VISIT…

COMPUTER-AIDED DESIGN,
ENGINEERING, AND MANUFACTURING
SYSTEMS TECHNIQUES AND APPLICATIONS

VOLUME
II

COMPUTER-INTEGRATED MANUFACTURING

© 2001 by CRC Press LLC

COMPUTER-AIDED DESIGN,
ENGINEERING, AND MANUFACTURING
SYSTEMS TECHNIQUES AND APPLICATIONS

VOLUME
II

COMPUTER-INTEGRATED MANUFACTURING

EDITOR
CORNELIUS LEONDES

CRC Press
Boca Raton London New York Washington, D.C.

Library of Congress Cataloging-in-Publication Data

Catalog record is available from the Library of Congress.

This book contains information obtained from authentic and highly regarded sources. Reprinted material is quoted with permission, and sources are indicated. A wide variety of references are listed. Reasonable efforts have been made to publish reliable data and information, but the author and the publisher cannot assume responsibility for the validity of all materials or for the consequences of their use.

Neither this book nor any part may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, microfilming, and recording, or by any information storage or retrieval system, without prior permission in writing from the publisher.

The consent of CRC Press LLC does not extend to copying for general distribution, for promotion, for creating new works, or for resale. Specific permission must be obtained in writing from CRC Press LLC for such copying.

Direct all inquiries to CRC Press LLC, 2000 N.W. Corporate Blvd., Boca Raton, Florida 33431.

Trademark Notice: Product or corporate names may be trademarks or registered trademarks, and are used only for identification and explanation, without intent to infringe.

© 2001 by CRC Press LLC

No claim to original U.S. Government works
International Standard Book Number 0-8493-0994-8
Printed in the United States of America 1 2 3 4 5 6 7 8 9 0
Printed on acid-free paper

Preface

A strong trend today is toward the fullest feasible integration of all elements of manufacturing including maintenance, reliability, supportability, the competitive environment, and other areas. This trend toward total integration is called concurrent engineering. Because of the central role information processing technology plays in this, the computer has also been identified and treated as a central and most essential issue. These are the issues which are at the core of the contents of this volume.

This set of volumes consists of seven distinctly titled and well-integrated volumes on the broadly significant subject of computer-aided design, engineering, and manufacturing: systems techniques and applications. It is appropriate to mention that each of the seven volumes can be utilized individually. In any event, the great breadth of the field certainly suggests the requirement for seven distinctly titled and well-integrated volumes for an adequately comprehensive treatment. The seven-volume titles are:

1. Systems Techniques and Computational Methods
2. Computer-Integrated Manufacturing
3. Operational Methods in Computer-Aided Design
4. Optimization Methods for Manufacturing
5. The Design of Manufacturing Systems
6. Manufacturing Systems Processes
7. Artificial Intelligence and Robotics in Manufacturing

The contributions to this volume clearly reveal the effectiveness and great significance of the techniques available and, with further development, the essential role that they will play in the future. I hope that practitioners, research workers, students, computer scientists, and others on the international scene will find this set of volumes to be a unique and significant reference source for years to come.

Cornelius T. Leondes
Editor

© 2001 by CRC Press LLC

Editor

Cornelius T. Leondes, B.S., M.S., Ph.D., emeritus professor, School of Engineering and Applied Science, University of California, Los Angeles has served as a member or consultant on numerous national technical and scientific advisory boards. He has also served as a consultant for numerous Fortune 500 companies and international corporations, published over 200 technical journal articles, and edited and/or co-authored over 120 books. He is a Guggenheim Fellow, Fulbright Research Scholar, and Fellow of IEEE, and is a recipient of the IEEE Baker Prize Award and the Barry Carlton Award of the IEEE.

© 2001 by CRC Press LLC

Contributors

Gabriella M. Acaccia
University of Genova
Genova, Italy

Massimo Callegari
University of Genova
Genova, Italy

Franco Caron
Politecnico di Milano
Milan, Italy

Marco Cristofari
Florida State University
Tallahassee, Florida

Stantcho N. Djiev
Technical University of Sofia
Sofia, Bulgaria

Sebastian Engell
Universität Dortmund
Dortmund, Germany

Maria Pia Fanti
Politecnico di Bari
Bari, Italy

Martin Fritz
Software Design & Management AG
Ratingen, Germany

Bruno Maione
Politecnico di Bari
Bari, Italy

Ernest L. McDuffie
Florida State University
Tallahassee, Florida

Rinaldo C. Michelini
University of Genova
Genova, Italy

Rezia M. Molfino
University of Genova
Genova, Italy

Andrew Y.C. Nee
National University of Singapore
Republic of Singapore

S.K. Ong
National University of Singapore
Republic of Singapore

Luben I. Pavlov
Technical University of Sofia
Sofia, Bulgaria

Giacomo Piscitelli
Politecnico di Bari
Bari, Italy

Roberto P. Razzoli
University of Genova
Genova, Italy

Stephen E. Sorensen
Hughes Information Technology Systems
Aurora, Colorado

Massimo Tronci
University of Rome "La Sapienza"
Rome, Italy

Biagio Turchiano
Politecnico di Bari
Bari, Italy

William J. Wolfe
University of Colorado at Denver
Denver, Colorado

Konrad Wöllhaf
Universität Dortmund
Dortmund, Germany

Swee-Hock Yeo
Nanyang Technological University
Republic of Singapore

Zhengxu Zhao
University of Derby
Derby, United Kingdom

© 2001 by CRC Press LLC

Contents

Chapter 1 System Approach to the Design of Generic Software for Real-Time Control of Flexible Manufacturing Systems (FMS)
Maria Pia Fanti, Bruno Maione, Giacomo Piscitelli, and Biagio Turchiano

Chapter 2 Computer-Integrated Assembly for Cost Effective Developments
Rinaldo C. Michelini, Gabriella M. Acaccia, Massimo Callegari, Rezia M. Molfino, and Roberto P. Razzoli

Chapter 3 Multi-Level Decision Making for Process Planning in Computer-Integrated Manufacturing (CIM) Systems
Zhengxu Zhao

Chapter 4 A Computer-Aided and Integrated Machining System
Swee-Hock Yeo

Chapter 5 Intelligent Set-Up Planning Systems for Parts Production in Manufacturing Systems
S.K. Ong and Andrew Y.C. Nee

Chapter 6 Object-Oriented Modeling and Simulation of Multipurpose Batch Plants
Sebastian Engell, Martin Fritz, and Konrad Wöllhaf

Chapter 7 Scheduling Systems and Techniques in Flexible Manufacturing Systems
Ernest L. McDuffie, Marco Cristofari, Franco Caron, Massimo Tronci, William J. Wolfe, and Stephen E. Sorensen

Chapter 8 Automated Systems Techniques for On-Line Quality and Production Control in the Manufacture of Textiles
Stantcho N. Djiev and Luben I. Pavlov

© 2001 by CRC Press LLC

1

System Approach to the Design of Generic Software for Real-Time Control of Flexible Manufacturing Systems (FMS)

Maria Pia Fanti
Politecnico di Bari

Bruno Maione
Politecnico di Bari

Giacomo Piscitelli
Politecnico di Bari

Biagio Turchiano
Politecnico di Bari

1.1 Preface
1.2 Dedication
1.3 Introduction
1.4 Fundamental Steps of the Modeling Process
1.5 Specification of the Model Static Structure
Workstations • Transport System
1.6 Queuing Network Configuration
Production Activity Configuration • Job Configuration • Server Configuration • Transport System Configuration • Definition of the Queuing Network Configuration
1.7 Scheduling Policy
1.8 The Discrete Event Dynamic System Model
Events • The DEDS Model
1.9 An Example of How the Modeling Process
The Model Static Structure of the Application Example • The Queuing Network Configuration of the Application Example • The Applied Scheduling Policy
1.10 The Generic Control Software Organization
The Control Hierarchy • Objects and Architecture of the Control Software
1.11 Conclusions
Appendix: List of Principal Symbols

© 2001 by CRC Press LLC

1.1 Preface

The control system is the core of any flexible manufacturing system (FMS) because it confers to the plant the capability to absorb internal changes and external fluctuations in demand and in production environment. However, the technical literature has repeatedly documented that poor control software design is the major source of difficulties in implementing FMSs. Namely, the FMS potentiality is not yet fully utilized, because typical, contemporary control software packages are still proprietary and do not possess flexibility and genericity. On the contrary, reducing the programming and reprogramming effort needs a generic software usable in an arbitrary FMS, producing an arbitrary part mix.

To design a generic software, two main problems must be solved. The former is to define an abstract formalism representing both hardware/software components of the FMS and the production plans. The latter consists in identifying a modular architecture of the control software capable of integrating standard modules.

To solve the first problem, we use a system approach leading to a discrete event dynamic system (DEDS) that constitutes a formal framework providing information on both the current operating conditions of the shop floor (SF) and the production progress. To face the second problem, we use the object-oriented approach to design the kernel of the generic control software. The resulting modular architecture is organized so that modules and interfaces among them can be replaced without altering the overall functionality.

1.2 Dedication

To our parents
Maria Pia Fanti, Biagio Turchiano
To the memory of my parents
Bruno Maione

1.3 Introduction

In recent years, the worldwide competition and the growing demand for a high variety of products have emphasized the role of production systems. In particular the FMSs are of increasing importance because they have the capability to face changes in production environment and to combine the efficiency of the mass-production lines with the flexibility of job-shops producing a variety of products in small- or medium-size batches. In any FMS, control software plays a key role since it can allow optimization of both hardware (numerically controlled machines, material handling devices, buffers, etc.) and software (information flow, data base content, etc.). Moreover FMS control should offer adaptive and dynamic features to absorb internal and external changes and to use all the resources efficiently. However, the literature has repeatedly documented that information flow and control software are the primary sources of difficulties in implementing FMSs. Namely, as a result of their proprietary nature, typical contemporary software packages for controlling manufacturing systems suffer from lack of flexibility and of genericity. They enable particular productivity improvements and focus on particular aspects of a specific plant. On the contrary, a generic software is necessary—usable in an arbitrary FMS producing an arbitrary part mix and minimizing the programming and reprogramming effort.

To design a generic and flexible control software, two main problems emerge. The first one is that effective control of the FMS activities depends on the existence of a formalism consisting of a unifying abstraction upon which a comprehensive and consistent knowledge base can be constructed. For this reason, the search for appropriate frameworks, taking into account the system operation, underlies recent efforts in the area of flexible automation Naylor and Maletz [14], Naylor and Volts [15]. The second problem consists in identifying an appropriate, modular software architecture capable of integrating standard software modules, e.g., for data base management, for field-events detection, and so on. Moreover, if there is the necessity, the modules of this architecture should be changeable to take into account specific instances of particular FMSs without changing the software main structure and interfaces among different parts.

© 2001 by CRC Press LLC

To solve the first problem, we refer to Zeigler's approach [20], [21], [22] which describes DEDS by using a set-theoretic formalism based on system theory concepts of Zadeh and Desoer [19], Wymore [18], and many others. In our approach the DEDS state encompasses information on both the current operating conditions of the SF and the progress of the programmed production. Moreover it includes a logic component collecting the decision rules necessary to resolve conflicts arising in the resource allocation. The information on the SF operating conditions and on the production progress is described using the primitive concepts of entities, entity attributes, and relationships between entities themselves, all organized in a relational structure [2] updated at each event occurrence. The decision rules, appearing as system state components, can be changed at any instant. Due to the formal structure of these rules, one can establish the information necessary to support the decision making.

The approach followed to define the DEDS state is transparent and has some advantages. First of all, it allows a detailed and modular description of the production system. Major details can be given, if necessary, by enriching the state with further entities, attributes, and relationships. Moreover, the model allows us to describe a generic FMS, producing an arbitrary part mix, under an arbitrary scheduling policy. Finally, the mechanism for state updating and the definition of all the exogenous events triggering such transitions lead to a DEDS model that completely describes the FMS behavior at job release and flow management levels.

To face the second problem, we consider the operation control of FMS from a model reference point of view. Our approach consists in driving the SF so that it follows the dynamics of the reference DEDS model. Indeed, this model establishes how the plant should respond to the command signals and interacts both with the higher level (production planning) and with the lower one (hardware components). The resulting control architecture emerges as direct consequence of this idea of utilizing the DEDS formulation to define the kernel of a generic software driving the dynamics of the actual FMS. Inspired by the works of Naylor and Volts [15], Naylor and Maletz [14], the paper [7] considerably improves and generalizes the formalism proposed in [5], necessary to develop a generic control software at the job release and flow management levels. Moreover, in the same line of [7], we emphasize the principles of structure, orderly partitioning, and modularity by using an object-oriented approach to design the control software. Namely, we decompose the control software into modules (objects) which encapsulate both data and procedures (methods) characterizing their behavior and which interact by sending one another messages. In addition the role and the interfaces of such objects remain unchanged, even if the FMS modifies its configuration to fit particular production needs. These characteristics are key features when the reuse of existing software (i.e., for planning, hardware components for managing data base, etc.) is one of the main objectives.

The organization of this contribution is as follows. The general methodology for building the sequential state of the DEDS model is the main concern of Section 1.4. Sections 1.5 and 1.6 specify the basic entities, relationships, and attributes defined on the entity sets. Section 1.7 introduces the formal representation of the decision rules that govern the flow of the transitional entities. Section 1.8 completes the DEDS model by defining events, DEDS state, model clock and mechanism for event-driven state transitions. Section 1.9 applies our formalism to a case study. Section 1.10 discusses the organization of a generic control software, as it emerges from the DEDS formulation. Finally, Section 1.11 draws the conclusions.

1.4 Fundamental Steps of the Modeling Process

To build an abstract comprehensive framework, we must specify the circumstances and restrict the aspects under which we consider the system. In other terms, we have to define an experimental frame. In respect to this fundamental point, we observe that even if the real behavior of the manufacturing plant involves variables changing continuously over time, the DEDS model abstracts from this behavior and records the occurrence of certain discrete events only. Namely, we consider the evolution of the manufacturing plant as a record (or a trace) of the occurrence of certain discrete, qualitative changes in the system, and ignore continuously occurring micro-changes. This is because we can describe the behavior of an FMS by means

© 2001 by CRC Press LLC

of sentences as "at instant t_1 the machine m_1 starts processing job j_2" or "at instant t_1 machine m_1 fails" or, again, "job j_1 enters the buffer b_1 which becomes empty," etc. Expressions building such sentences e.g., "starts processing part," "the machine fails," "the job enters the buffer," "the buffer becomes empty," etc. indicate the occurrence of discrete events, but ignore continuous changes in variables such as "the amount of metal cut," "the raw material consumption," "the instantaneous speed of the transport units," etc.

In conclusion, we model a generic FMS taking into account sudden changes of states occurring at instants determined by the internal system dynamics or by the environment actions. In contrast with conventional dynamic systems, however, states have logical or symbolic, rather than numerical, values which change on the occurrence of events. To establish this framework, we refer to the concepts used by Zeigler [20], [21], [22] by developing an appropriate DEDS formalism based on the expressive and rigorous tools of logic and set theory.

The proposed approach builds the FMS model in a hierarchical, modular manner by putting parts together to form larger ones in successive steps.

The first step is to identify the basic entities of the framework. These are both hardware and software elements and may, or not, change with time. Homogeneous entities are grouped in appropriate sets (entity sets) which, on their part, belong to three categories:

1. The resource entity sets collect the basic elements providing service for customers flowing through the network of queues. Typically, this class includes machine, buffer, server, and transport entity sets. The cardinality and the members of such sets are permanent elements of the model structure even if some resource is unavailable occasionally.
2. The processing activity entity sets contain the basic elements of the manufacturing activity. Sets belonging to this category consist of orders, job types, active processing procedures, and allowable alternative sequences of operation steps in a given working procedure. The above entities change during the manufacturing process and hence are transitionary.
3. The job entity set contains the basic temporary entities of the model, i.e., jobs (pieces) in the system at a given time. Both cardinality and elements of such a set change with time.

The second step of the procedure deals with establishing associations between entity sets and with defining entity attributes. Associations take the form of mathematical relations among entities and are expressed as functions mapping from an entity set into another one or appear in the form of specified relationship sets. Functions mapping from an entity set (or a relationship set) into different value sets (such as the set of reals, non-negative integers, {0, 1}, {*Up*, *Down*}, etc.) define the entity (relationship) attributes. Typical attributes encountered in the proposed framework are the number of final products, the buffer capacity, the server status, the current position of an Automated Guided Vehicle (AGV), etc. Attributes either characterize each of the entities in the system or keep track of their current status.

An entity may also be indirectly related to another one. Hence the need arises of an overall perspective in which all the entities, together with their attributes and relationships, contribute to represent the FMS as a whole. This leads to the concept of configuration. A configuration collects all the relationships and the attributes defined on entity sets (or subsets) belonging to the same category. For instance, the job entity set (or a subset of it) is the domain of attributes and relationships pertaining to the job configuration ($\mathbf{C_j}$). Furthermore both the server configuration ($\mathbf{C_s}$) and the transport system configuration ($\mathbf{C_h}$) consist of relationships and attributes defined on resource entity sets (subsets). Finally the production activity configuration ($\mathbf{C_p}$) includes all the attributes and relationships defined on processing activity entity sets, e.g., job type, order, working procedure, operation step entity sets, etc. The 4-tuple ($\mathbf{C_p, C_j, C_s, C_h}$), named queuing network configuration, embodies all the entities with their attributes and relations integrating all the elements in a whole.

The third step provides the formal representation of the set of decision rules governing the flow of the transitional entities. Rules concern job loading, routing, and priority setting. In the proposed framework, the focus is on the formal representation of rules rather than on the rules themselves. Namely, the sole information about a rule is how it utilizes the queuing network configuration knowledge to make proper decisions.

© 2001 by CRC Press LLC

The fourth step defines the model clock, i.e., the event occurrence times, and measures the elapsed time from the last event occurrence. We consider two types of events. The internal system dynamics generates the internal events (operation step and transport operation completion, etc.). The external events (hardware component failure or repair, order removal, or insertion, etc.) represent disturbances or manipulated variables. In a time interval in which no event takes place, ($\mathbf{C_p}$, $\mathbf{C_j}$, $\mathbf{C_s}$, $\mathbf{C_h}$) remains unchanged.

All four steps lead to the set-theoretic foundation of the DEDS model.

1.5 Specification of the Model Static Structure

This section presents the structural view underlying the model. It combines entities, relationships and attributes that do not change with time and describe the system resources. Typical entities are machines, buffers, servers, and transport subsystems. Typical relations express association between resources. Typical attributes define resource capacity.

Workstations

We begin to characterize the FMS workstations by defining the machine, buffer, and server entity sets.

Consider an FMS with N_M available workcenters. These constitute the set

$$\text{Machine entity set: } M = \{m_i, i = 1, 2, \ldots, N_M\}$$

Each machine m_i is provided with an input/output buffer, or, alternatively, with both an input buffer and an output one. Sometimes, the FMS has a central buffer where jobs wait for their next destination. At this point we introduce the set

$$\text{Buffer entity set: } B = \{b_\mu, \mu = 1, 2, \ldots, N_B\}$$

where N_B indicates the number of buffers in the system. The attribute

$$\text{TB: } B \rightarrow \{\text{I, O, IO}\}$$

distinguishes the type of buffer, where $\text{TB}(b_\mu) = \text{I(O, IO)}$ indicates that b_μ is an input (output, input/output) buffer. The central buffer can be viewed as the IO buffer of a fictitious machine (say m_0) that must be included in the set M. Moreover, the buffer capacity attribute

$$\text{BC: } B \rightarrow \text{I}^+$$

assigns a finite size to each buffer. Here I^+ is the positive integer set.

Finally, each machine m_i is equipped with S_i servers. All the servers within the system make up the

$$\text{Server entity set: } S = \{s_{ij}, i = 1, 2, \ldots, N_M, j = 1, 2, \ldots, S_i\}$$

Now, to specify how elements from B (from S) are related to elements of M, we introduce the functions

$$\text{BEL-}B\text{: } B \rightarrow M$$
$$\text{BEL-}S\text{: } S \rightarrow M$$

where $\text{BEL-}B(b_\mu) = m_i$ indicates that the buffer b_μ belongs to m_i and $\text{BEL-}S(s_{ij}) = m_i$ means that s_{ij} is a server of m_i. The fictitious machine has no server.

© 2001 by CRC Press LLC

Transport System

Within the FMS, there are many possible transport subsystems to transfer pieces from a machine to another one. Here we consider only AGV systems, while [7] describes a wide variety of transport systems.

So let N_H AGV subsystems be available. An AGV subsystem h_n ($n = 1, 2, \ldots, N_H$) consists of N_n trucks (AGV units). For sake of simplicity, we assume that each truck can carry only one piece at a time from the output (input/output) buffer of a machine to the input (input/output) buffer of another workcenter. We group the transport subsystems in a specific entity set

$$\text{Transport subsystem entity set: } H = \{h_n, n = 1, 2, \ldots, N_H\}$$

Analogously, we arrange the trucks in the following set

$$\text{Transport unit entity set: } U = \{u_{nr}, n = 1, 2, \ldots, N_H, r = 1, 2, \ldots, N_n\}$$

The following function

$$\text{BEL-}U: U \rightarrow H$$

associates each element from the transport unit entity set with the corresponding one in the transport subsystem entity set and the relationship set (transport set)

$$T = \{(h_n, m_k, m_j) \mid h_n \in H \quad m_k, m_j \in M\}$$

specifies all the transports the material handling system is able to perform. Finally the function

$$\text{TT}: T \rightarrow \text{R}^+$$

expresses the transport time. In particular, $(h_n, m_k, m_j) \in T$ if the subsystem h_n can carry a piece from m_k to m_j, while $\text{TT}[(h_n, m_k, m_j)]$ indicates the time such a transport requires.

1.6 Queuing Network Configuration

The sequential state specification still requires further entities, relationships, and attributes organized in four configurations and constituting the queuing network configuration. In contrast to the structural view, the queuing network configuration varies with time, tracing the system dynamic path.

Production Activity Configuration

To keep track of the current status of the production activities, we have to take full account of shop orders by specifying their due dates, item types, and quantities. Moreover we must describe the working procedures of each job type and specify how and how many parts are assembled to compose final products. Finally, since many alternative sequences of operation steps can make up the same working procedure, we must indicate the alternative routes each job can follow, the machines involved, and the time each operation step requires.

In the sequel we describe orders, working procedures, and operation steps separately. However, they all contribute to set up the production activity configuration.[12]

© 2001 by CRC Press LLC

Production Orders

An FMS may manufacture final products of N_P types, grouped in N_O production orders. So the set

$$\text{Job type entity set: } P = \{p_\pi, \pi = 1, 2, \ldots, N_P\}$$

identifies job types (final product types) in the production mix.

A production order is a demand for the production of one or more types of pieces.

The set

$$\text{Order entity set: } O = \{o_\gamma, \gamma = 1, 2, \ldots, N_O\}$$

collects the identifiers of the orders currently in execution and the function

$$\text{BEL-}P\text{: } P \rightarrow O$$

specifies the production order corresponding to each piece type.

To complete the description of a production order, we specify the attributes due date (DD) and the product number (PN) demanded for each type

$$\text{DD: } O \rightarrow \mathrm{R}^+$$
$$\text{PN: } P \rightarrow \mathrm{I}^+$$

Job Working Procedures

A final product may result from assembly operations. Hence, the need arises to establish the way of combining intermediate products. To this end, we preliminary denote with W_π the number of working procedures necessary to obtain the P_π-type final product. All the working procedures corresponding to all the job types build up the

$$\text{Working procedure entity set: } W = \{w_{\pi\alpha}, \pi = 1, \ldots, N_P, \alpha = 1, \ldots, W_\pi\}$$

Moreover the function

$$\text{BEL-}W\text{: } W \rightarrow P$$

indicates that every working procedure is in relation with a job type.

At this point we are ready to introduce the following assembly relationship set

$$A = \{(w_{\pi\alpha}, w_{\pi\beta}) \,|\, w_{\pi\alpha}, w_{\pi\beta} \in W \quad \text{BEL-}W(w_{\pi\alpha}) = \text{BEL-}W(w_{\pi\beta}) = \mathrm{p}_\pi\}$$

which describes the ways of assembling intermediate into finished products. The pair $(w_{\pi\alpha}, w_{\pi\beta}) \in A$ iff (if and only if) the parts resulting from the working procedure $w_{\pi\alpha}$ are components of items manufactured according to $w_{\pi\beta}$. The relationship set A has the attribute weight

$$\text{WE: } A \rightarrow \mathrm{I}^+$$

where $\mathrm{WE}(w_{\pi\alpha}, w_{\pi\beta})$ defines the number of pieces resulting from $w_{\pi\alpha}$ which concur in the assembly of one item to submit to $w_{\pi\beta}$.

Finally, the subset W_I collects elements from W representing initial working procedures (to perform on not assembled jobs just loaded into the system).

© 2001 by CRC Press LLC

Operation Steps

A working procedure is executed by a sequence of operation steps selected in a set of alternative allowable sequences. Denoting by $V_{\pi\alpha}$ the number of operation steps composing $w_{\pi\alpha}$, we introduce the

$$\text{Operation step entity set: } V = \{v_{\pi\alpha\beta},\ \pi = 1,\ldots,N_P \quad \alpha = 1,\ldots,W_\pi \quad \beta = 1,\ldots,V_{\pi\alpha}\}$$

The relationship

$$\text{BEL-}V\text{: } V \rightarrow W$$

indicates the working procedure which an operation step belongs to and the following function

$$\text{PT: } V \rightarrow \mathrm{R}^{+}$$

represents the attribute processing time of the operation steps.

Furthermore, the alternative routing relationship describes all the alternative sequences carrying out a working procedure

$$R = \{(v_{\pi\alpha\beta}, v_{\pi\alpha\delta}) \mid v_{\pi\alpha\beta}, v_{\pi\alpha\delta} \in V \quad \text{BEL-}V(v_{\pi\alpha\beta}) = \text{BEL-}V(v_{\pi\alpha\delta})\}$$

The pair $(v_{\pi\alpha\beta}, v_{\pi\alpha\delta}) \in R$ iff there exists an allowable sequence in which the operation step $v_{\pi\alpha\beta}$ is right before $v_{\pi\alpha\delta}$.

Since any operation step may require one or more machines (co-operating machines), we associate an ordered couple $(m_l, \underline{M}_l)$ with each $v_{\pi\alpha\beta}$, where $m_l \in M$ stands for the machine hosting the piece, while $\underline{M}_l \subseteq M - \{m_l\}$ defines the set of workcenters co-operating with m_l. If $\{\underline{M}\}$ is the set of all the subsets of M with cardinality k ($k = 0, 1, \ldots, N_M - 1$), the job hosting machine (JHM) and the co-operating machines (COM) functions establish the following relationships between the entity sets V and M

$$\text{JHM: } V \rightarrow M$$
$$\text{COM: } V \rightarrow \{\underline{M}\}$$

Here $\text{JHM}(v_{\pi\alpha\beta})$ and $\text{COM}(v_{\pi\alpha\beta})$ specify respectively the first and the second element of the couple $(m_l, \underline{M}_l)$ associated with $v_{\pi\alpha\beta}$. Of course, if $\underline{M}_l$ is the empty set, the operation step $v_{\pi\alpha\beta}$ needs machine m_1 only.

Finally, the attribute type of step

$$\text{TS: } V \rightarrow \{\mathrm{N}, \mathrm{A}\}$$

indicates if any operation step is normal (N) or involves assembling of parts (A).

For systems with a central buffer a particular remark is in order. Namely in this case each of sets V and W must contain a special element (respectively v_0 and w_0) with $\text{BEL-}V(v_0) = w_0$, $\text{PT}(v_0) = 0$, $\text{TS}(v_0) = N$, $\text{JHM}(v_0) = m_0$ and where $\text{COM}(v_0)$ is equal to the empty set. The operation step v_0 does not stand for a real processing operation, but it indicates that a job is at m_0, i.e., in the central buffer. Obviously, v_0 does not pertain to any "true" working procedure.

The 16-tuple

$$C_p = \{O, P, W, A, V, R, DD, BEL\text{-}P, PN, BEL\text{-}W, WE, BEL\text{-}V, PT, JHM, COM, TS\}$$

is defined as the "Production Activity Configuration" and may change with time, for example in consequence of a modification in the production orders.

© 2001 by CRC Press LLC

Job Configuration

In our experimental frame, when a job enters the system, it lies in the buffer of the first machine that must process it. If the first operation step can take place immediately, the job directly goes on to one of the machine servers with no delay. If all the servers of the workstation are busy, the piece waits in the input (input/output) buffer. Once a server becomes free, the job engages it and the machining process can start. In some cases more servers of different machines must concur to the machining. After one or more activity completions, the job goes on to the output (input/output) buffer, waiting for a new destination, according to a selected route of the associated working procedure.

However there are many problems that make the modeling more difficult. First, the machine input buffer is in general not empty because many jobs may be queuing. If the buffer is completely full it cannot receive more pieces. In this case the transport unit cannot unload the job and remains blocked. Finally, after the completion of machining, the job cannot immediately enter the output buffer if this is full. In this case, the job remains blocked on the server of the hosting machine.

After these preliminary remarks, we come to our concern. If N_J is the number of pieces currently in the system and J is the corresponding

$$\text{Job entity set: } J = \{j_k, k = 1, 2, \ldots, N_J\}$$

the type of job function

$$\text{TJ: } J \rightarrow P$$

states the relationship between pieces and job types.

The following function defines the attribute specifying the operative status of a job

$$J\text{-STATUS: } J \rightarrow \{\textit{Ready, Ready for assembling, Running, Waiting for destination Waiting for transport, Carried}\}$$

where J-STATUS(j_k) equals *Ready*, if j_k is in an input or input/output buffer and not all the required servers are available; *Ready for assembling*, if j_k is in a buffer and the pieces concurring in the assembly are not all available; *Running*, if j_k is receiving service from one or more machines or if, after an operation completion, it cannot move to the output buffer, because the hosting server is blocked; *Waiting for destination*, if j_k resides in an output or input/output buffer and is waiting for the selection of its next destination and of an appropriate transport unit; *Waiting for transport*, if j_k is in an output or input/output buffer and is waiting for a transport unit. In this case, both the next operation step and the transport unit have been already chosen; and *Carried*, if j_k is moving to the next workcenter.

According to the values of J-STATUS(j_k), we can partition the job entity set into the following six subsets: J_R, J_{RA}, J_{Ru}, J_{DW}, J_{TW}, and J_C. Indeed, introducing the above subsets is necessary to define the functions pertaining to the job configuration. A first group of functions associates jobs with entities belonging to the processing activity entity sets. Namely, at the instant t each piece within the system requests or receives service, according to an operation step from V. This relation is indicated by the function reference operation step:

$$\text{ROS: } J \rightarrow V$$

© 2001 by CRC Press LLC

Depending on the value of J-STATUS(j_k), the previous function takes the following meanings

if $j_k \in J_R \cup J_{RA} \cup J_{TW} \cup J_C$, $ROS(j_k)$ specifies the next operation step already assigned to j_k;
if $j_k \in J_{Ru}$, ROS(j_k) coincides with the operation step in progress;
if $j_k \in J_{DW}$, ROS(j_k) specifies the operation step just completed.

In a similar way the function last operation step

$$\text{LOS: } J \rightarrow V \cup \{\text{n.o.s.}\}$$

indicates the operation step executed by a job just before ROS(j_k). The condition LOS(j_k) = n.o.s. (no operation step) means that ROS(j_k) is the first operation step executed by (or to execute on) j_k.

A second group of functions maps job entity subsets into the resource entity sets. In particular, the following functions establish relationships between subsets of J and the buffer entity set and the transport unit entity set

$$\text{HOS: } J_R \cup J_{RA} \cup J_{DW} \cup J_{TW} \rightarrow B$$
$$\text{BOO: } J_{TW} \rightarrow U$$
$$\text{CAR: } J_C \rightarrow U$$

where HOS(j_k) indicates the buffer hosting j_k, BOO(j_k) is the transport unit booked for transferring j_k to its next destination and CAR(j_k) denotes the transport unit transferring j_k to the destination workcenter.

An important attribute in the DEDS model framework is the time to complete either a processing (if $j_k \in J_{Ru}$) or a transport operation (if $j_k \in J_C$). The "Residual Operation Time" function

$$\text{ROT: } J \rightarrow \mathrm{R}^+ \cup \{\infty\}$$

specifies this attribute. In particular ROT(j_k) defines the residual time required for the booked transport (i.e., BOO(j_k)) to reach the buffer hosting j_k, if $j_k \in J_{TW}$ and the whole time to perform the next operation step (i.e., ROS(j_k)), if $j_k \in J_R \cup J_{RA}$; finally ROT(j_k) = ∞, if $j_k \in J_{DW}$. Note that each residual time is always evaluated starting from the instant of the last event occurrence.

As mentioned before, blocking conditions may occur during the FMS operation, i.e., when, on a transport or an operation step completion, the destination buffer is full. In such a case, the job remains blocked on the transport unit or on the hosting machine server. The blocking attribute function

$$\text{BA: } J_{Ru} \cup J_C \rightarrow \{0, 1\}$$

points out this condition. Namely BA(j_k) equals 1 (0) if j_k is blocked (not blocked). There is another situation similar to blocking, indicating that a job is prevented from changing its status. In this case the job cannot release the resource it currently holds. This condition can be indicated by the attribute "State Change Inhibitor"

$$\text{SCI: } J \rightarrow \{0, 1\}$$

where SCI(j_k) = 1 means that J-STATUS(j_k) is prevented from changing value. On the contrary, if SCI(j_k) = 0 no limitation is put on the modification of J-STATUS(j_k).

Finally, further attributes can be introduced that are useful to take job priority decisions. Some examples are the arrival time at the buffer and the arrival time at the system

$$\text{BAT: } J_R \cup J_{RA} \cup J_{DW} \cup J_{TW} \rightarrow \mathrm{R}^+$$
$$\text{SAT: } J \rightarrow \mathrm{R}^+$$

© 2001 by CRC Press LLC

Now we are in the position to define the 13-tuple:

C_j = {*J*, TJ, *J*-STATUS, ROS, HOS, BOO, CAR, ROT, LOS, BA, SCI, BAT, SAT} named job configuration. It fully characterizes the jobs within the system and the operations which they are currently subject to.

Server Configuration

The number of available servers may change with time as a result of occasional failures and repairs. To exhibit the server operative conditions, we introduce the server status attribute

$$\text{S-STATUS}: \text{S} \rightarrow \{\textit{Idle}, \textit{Running}, \textit{Blocked}, \textit{Down}, \textit{Reserved}\}$$

where *S*-STATUS(s_{ij}) is equal to: *Idle*, if s_{ij} is inactive; *Running*, if s_{ij} is processing a piece; *Blocked*, if s_{ij} cannot unload the piece already processed because the output (input/output) buffer is full; *Down*, if s_{ij} is out of order; *Reserved*, if s_{ij} is not currently busy at service, but it is booked for some operation step requiring machine co-operation.

In particular, if *S*-STATUS(s_{ij}) = *Reserved*, the job requiring service from s_{ij} is put in relation to such a server. Analogously, *Running* or *Blocked* servers are in relation to *Running* jobs. Such relationships are specified by the functions reserve and process

$$\text{RES}: \; S_R \rightarrow J_R \cup J_{RA}$$
$$\text{PRO}: \; S_{RuB} \rightarrow J_{Ru}$$

where the subset S_R collects the *Reserved* servers, while S_{RuB} groups the *Running* or *Blocked* ones. Finally the triple

$$C_s \; = \; \{S\text{-STATUS, RES, PRO}\}$$

is called the server configuration.

Transport System Configuration

The following function characterizes the status of each truck

$$U\text{-STATUS}: \; U \rightarrow \{\textit{Idle}, \textit{Carrying}, \textit{Blocked}, \textit{Booked}, \textit{Down}\}$$

with an obvious meaning of the four conditions. In particular, *U*-STATUS(u_{nr}) = *Booked* if u_{nr} is moving to the buffer hosting the piece j_k to transport next. When *U*-STATUS(u_{nr}) = *Idle*, we assume that u_{nr} is standing close to a machine. So, denoting by U_{Id} the subset of *Idle* trucks, we establish such a relationship by the function rest close to the machine

$$\text{RCM}: \; U_{Id} \rightarrow M$$

To account for failure conditions of a whole transport subsystem, we introduce the Transport Subsystem status attribute

$$H\text{-STATUS}: \; H \rightarrow \{\textit{Up}, \textit{Down}\}$$

In conclusion, the triple

$$C_h \; = \; \{U\text{-STATUS, RCM, } H\text{-STATUS}\}$$

fully describes the transport system configuration.

© 2001 by CRC Press LLC

Definition of the Queuing Network Configuration

We call queuing network configuration the 4-tuple $\mathbf{F} = (\mathbf{C_p}, \mathbf{C_j}, \mathbf{C_s}, \mathbf{C_h})$ and denote by $\mathbf{F}(t)$ its value at time t. Moreover, F indicates the set of all the 4-tuples $\mathbf{F}$. Using standard symbols introduced in [13], Figure 1.1 assembles all the entities and the relationships composing both queuing network configuration and structural view.

1.7 Scheduling Policy

Adding flexibility to the factory floor increases the scheduling alternatives and, hence, the decision complexity. Moreover, the necessity of quickly reacting to dynamic changes in the system makes the operation scheduling more difficult. To approach this problem, we consider the currently applied scheduling rules as an essential component of the system state. That is, at each instant, the rules are elements of the knowledge of the current system behavior. Hence, we do not propose any particular scheduling policy for resource management at the job release and job flow levels; rather we decompose the scheduling problem into elementary decision makings (micro-decisions) generally concerning the selection of one entity (job, operation step, server transport unit, etc.) in a proper subset. Then, to formally describe such micro-decisions, we introduce appropriate functions (decision rules) by indicating their domain and their range and leaving undetermined their specific decision mechanism. In essence a decision rule corresponds to a module that, when invoked, receives data from the current queuing network configuration and makes the decisions it is in charge of. Furthermore a scheduling policy is a set of decision rules, each of which specifies a micro-decision. In consequence of a modification in production goals, the scheduling policy may vary with time because at any instant the upper decision levels may require to apply a new rule and to remove another one. To this aim, a set of possible scheduling rules can be predetermined by the planning staff and stored in a library.

The job loading and job flow management require the choice of the product type to introduce next into the system and the corresponding loading time, the routing of each job within the systems, if alternative paths are available, and the resource allocation when concurrence conditions arise.

Job loading decision rules. Decisions concerning the piece type to introduce next into the system and the corresponding loading time must be taken simultaneously. The sequencing function and timing function describe the loading decisions

$$\omega_{ls}: F \rightarrow W_I \cup \{\text{n.c.}\}$$
$$\omega_{lt}: F \rightarrow R^{+} \cup \{\infty\}$$

In particular, $\omega_{ls}(\mathbf{F})$ specifies the initial working procedure of the next job entering the system. In other words, the value of $\omega_{ls}(\mathbf{F})$ particularizes the raw piece to load and $\omega_{lt}(\mathbf{F})$ indicates the interval between the current time and the loading instant. The trivial element "n.c." stands for "no choice is possible," so that when $\omega_{ls}(\mathbf{F})$ equals "n.c.," $\omega_{lt}(\mathbf{F})$ must be ∞. This indicates that no choice is possible, till a change in the 4-tuple $\mathbf{F}$ occurs. By assumption, if $\omega_{ls}(\mathbf{F}) \neq$ n.c., a new job can enter the system and receive service according to $\omega_{ls}(\mathbf{F})$. In this case, the first buffer that has to host the new piece is not full.

State change inhibition rule. On an event occurrence, it establishes if the status change of a job is allowed or inhibited

$$\omega_{sci}: F \times J \rightarrow \{0, 1\}$$

In particular, $\omega_{\text{sci}}(\mathbf{F}, j_k) = 1$ (=0) indicates that SCI(j_k) must be set to 1 (0).

Job routing decision rules. As mentioned in Section 1.6, a job is processed according to a sequence of operation steps selected from a set of alternatives. In particular such choices concern the following.

© 2001 by CRC Press LLC

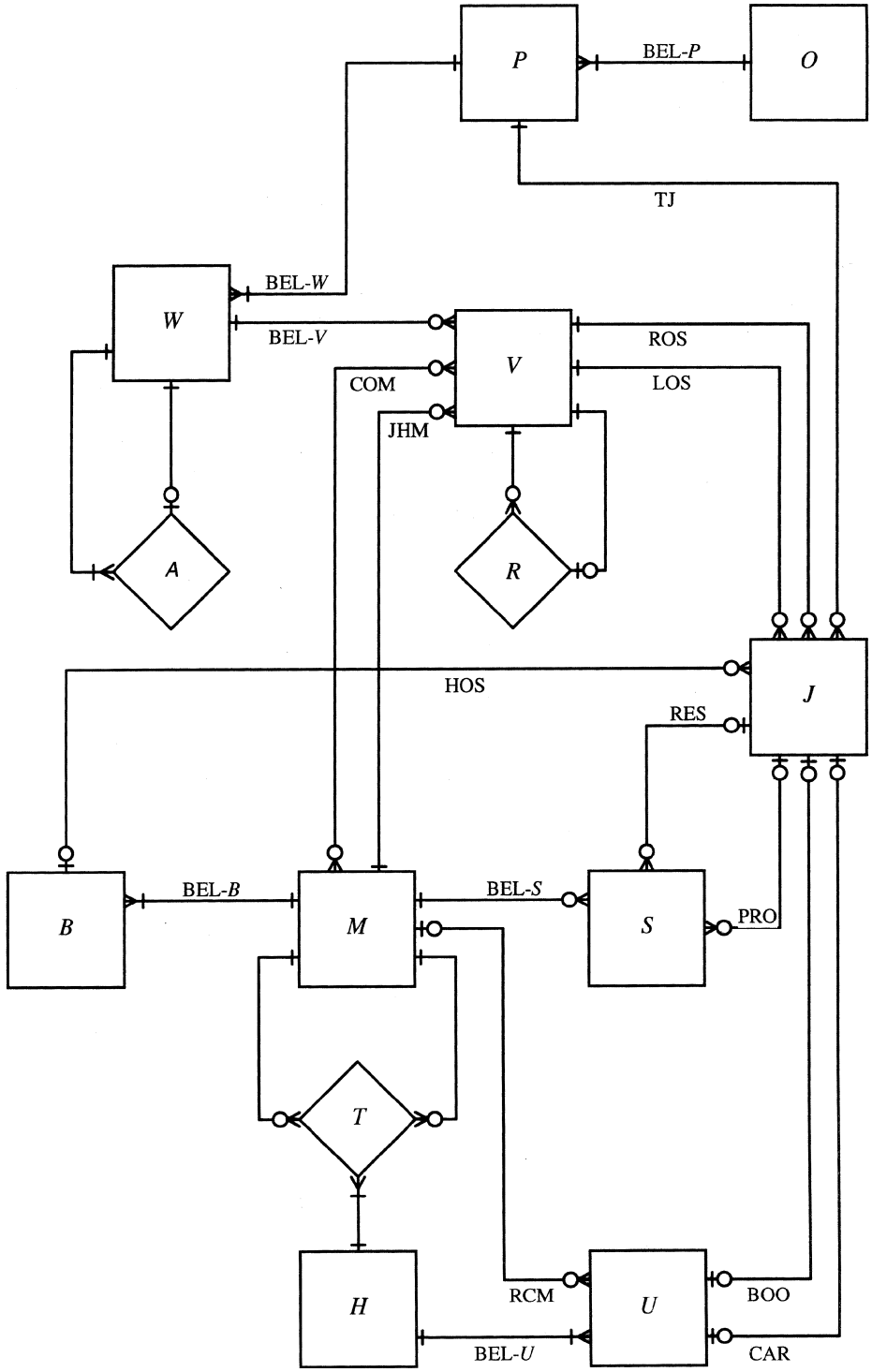


FIGURE 1.1 Entity-relationship diagram of both queuing network configuration and structural view.

© 2001 by CRC Press LLC

1. The next step among the alternative operations in a working procedure. The following decision function

$$\omega_v\colon\ F \times V \to V \cup \{\text{n.c.}\}$$

rules the selection. Here $\omega_v(\mathbf{F}, v_{\pi\alpha\beta})$ specifies the step to execute after the operation already completed (i.e., after $v_{\pi\alpha\beta}$). Obviously, we get $[v_{\pi\alpha\beta}, \omega_v(\mathbf{F}, v_{\pi\alpha\beta})] \in R$ iff $\omega_v(\mathbf{F}, v_{\pi\alpha\beta}) \neq$ n.c.
2. The first operation step in a working procedure. The corresponding ruling function is

$$\omega_{v0}\colon\ F \times W \to V$$

where $\omega_{v0}(\mathbf{F}, w_{\pi\alpha})$ yields the initial operation step selected to perform the working procedure $w_{\pi\alpha}$.

Priority decision rules. These decisions concern jobs, servers, and transport units. The choices regarding jobs are:

1. The selection of a piece among those waiting for receiving service from a machine. The corresponding ruling function is

$$\omega_j\colon\ F \times M \to J_R \cup \{\text{n.c.}\}$$

where $j_k = \omega_j(\mathbf{F}, m_i)$ indicates the next *Ready* piece, with $\text{SCI}(j_k) = 0$, that m_i has to process.
2. The selection of the next job that will use a transport unit. Function

$$\omega_{jh}\colon\ J \to J_{DW} \cup \{\text{n.c.}\}$$

makes this decision. In particular $\omega_{jh}(\mathbf{F})$ specifies a job with $\text{SCI}[\omega_{jh}(\mathbf{F})] = 0$ among the ones *Waiting for destination* in the output and input/output buffers.
3. The choice of a set of pieces that will concur in an assembly operation. If $\underline{J}_R$ denotes the set of all the subsets of J_R, the function

$$\omega_{a,R}\colon\ F \times J \to \underline{J}_R$$

rules this selection. In particular, $\omega_{a,R}(\mathbf{F}, j_k)$ indicates a set of *Ready* jobs such that $\{j_k\} \cup \omega_{a,R}(\mathbf{F}, j_k)$ collects all the noninhibited jobs necessary to perform the assembly operation step $\text{ROS}(j_k)$.
4. The selection of jobs required for an assembly operation, which can change their status from *Ready for assembling* to *Ready*. Function

$$\omega_{a,RA}\colon\ F \times J \to \underline{J}_{RA}$$

expresses this choice, where $\underline{J}_{RA}$ is the set of all the subsets of J_{RA}. In particular $\omega_{a,RA}(\mathbf{F}, j_k)$ gives a set of *Ready for assembling* and noninhibited jobs, such that $\{j_k\} \cup \omega_{a,RA}(\mathbf{F}, j_k)$ collects all the jobs necessary to perform the assembly step $\text{ROS}(j_k)$.
5. The selection of a job among those prevented from changing status. If $\underline{J}_{SCI}$ denotes the set of jobs with State Change Inhibitor equal to 1, the function

$$\omega_i\colon\ F \to \underline{J}_{SCI} \cup \{\text{n.c.}\}$$

chooses a job $j_k \in \underline{J}_{SCI}$ in order to reset to zero its state change inhibitor. The condition $\omega_i(\mathbf{F}) =$ n.c. means that the function does not select any job.

© 2001 by CRC Press LLC

The choices regarding servers refer to:

6. The selection of an idle server to perform an operation step. The function

$$\omega_s: F \times M \rightarrow S \cup \{\text{n.c.}\}$$

rules this decision. Hence $\omega_s(\mathbf{F}, m_i)$ identifies an *Idle* server belonging to m_i.

7. The selection of the first server to consider available for a new operation, among those completing an operation step. Let $\underline{S}$ be a server subset such that for any couple of distinct elements s_{i_1,j_1}, s_{i_2,j_2} from $\underline{S}$ it holds: BEL-S $(s_{i_1,j_1}) \neq$ BEL -S(s_{i_2,j_2}). Denoting by $\{\underline{S}\}$ the set of all possible $\underline{S}$, we introduce the function

$$\omega_{fs}: F \times \{\underline{S}\} \rightarrow S$$

where $\omega_{fs}(\mathbf{F}, \underline{S})$ defines a server chosen from S;

8. The selection of a server to unblock next. The ruling function is

$$\omega_{bs}: F \times M \rightarrow S \cup \{\text{n.c.}\}$$

with $\omega_{bs}(\mathbf{F}, m_i)$ specifying a *Blocked* server belonging to m_i.

The choices regarding transport units relate to:

9. The selection of a unit for a job transport. The function

$$\omega_h: F \times M \times M \rightarrow U \cup \{\text{n.c.}\}$$

expresses this decision where $\omega_h(\mathbf{F}, m_i, m_j)$ indicates the transport unit chosen to move a piece from m_i to m_j, with $\{\text{BEL-U}[\omega_h(\mathbf{F}, m_i, m_j)], m_i, m_j\} \in T$ if $\omega_h(\mathbf{F}, m_i, m_j) \neq$ n.c.

10. The selection of a transport unit to unblock next. The ruling function is

$$\omega_{bh}: F \times M \rightarrow U \cup \{\text{n.c.}\}$$

The unit $\omega_{bh}(\mathbf{F}, m_i)$ is chosen among the *Blocked* ones waiting for transferring the just carried piece to the input (input/output) buffer of m_i.

Finally, the 15-tuple:

$$\Omega = [\omega_{ls}, \omega_{lt}, \omega_{sci}, \omega_v, \omega_{v0}, \omega_j, \omega_{jh}, \omega_{a,R}, \omega_{a,RA}, \omega_i, \omega_s, \omega_{fs}, \omega_{bs}, \omega_h, \omega_{bh}]$$

defines the scheduling policy and $\Omega(t)$ denotes its value at time t.

Before concluding this section, we define the residual time to the next job input (RT), to complete the information on the system operating condition. Namely, since the timing function ω_{lt} is invoked at discrete times (i.e., at the event occurrences only), it is necessary to keep memory of the scheduled time for the next job input. In particular RT(t) is the residual time to the next job input evaluated from the instant of the last occurred event.

In conclusion, in our experimental frame the 3-tuple

$$\mathbf{C}(t) = [\mathbf{F}(t), \text{RT}(t), \Omega(t)]$$

holds the information on the system operating condition. $\mathbf{C}(t)$ is the sequential state at time t for the "DEDS model [21], [22]. In particular, the pair $[\mathbf{F}(t), \text{RT}(t)]$ corresponds to the informational component," i.e., to a set of structured data, describing in-progress activity plans ($\mathbf{C}_p$), status of both resources and jobs, their mutual associations and, finally, their relations with manufacturing activities ($\mathbf{C}_j$, $\mathbf{C}_s$, $\mathbf{C}_h$). The Scheduling Policy $\Omega(t)$ expressing the decision procedures is the second component of the sequential state.

© 2001 by CRC Press LLC

The need for the state to hold information on production activity plans stems from our point of view, that considers not only the SF world, but also the job release and flow management level. For the same reason, we consider the scheduling policy as a component of the sequential state.

1.8 The Discrete Event Dynamic System Model

Previous sections specify the information necessary to define the sequential state $\mathbf{C}(t)$ that consists of two components, different in meaning and characteristics.

The knowledge of both the informational component and the decision procedures contributes to determine the system future behavior, provided that the future external stimuli are also known. Such stimuli (i.e., the inputs to the DEDS model) correspond to events, i.e., to occurrences that modify the state of a DEDS significantly [20], [21], [22]. As mentioned in Section 1.4, we classify events in two distinct groups: the external events and the internal ones. External events represent the influence of the environment on the system, i.e., they specify the inputs to the model. The internal ones arise from the inner system dynamics.

Events

The types of events relevant to our experimental frame are the following: server failure or repair (e_1), AGV truck failure or repair (e_2), transport subsystem failure or repair (e_3), new production order insertion (e_4), production order removal (e_5), scheduling policy updating (e_6), loading of a new job (e_7), operation step completion (e_8), transport operation completion (e_9), and transport unit positioning completion (e_{10}).

Events from e_1 to e_6 are external. In particular, events from e_1 to e_3 play the role of disturbances, while those from e_4 to e_6 correspond to manipulated inputs, under the control of the FMS production planning manager. Since any external event requires a suitable unambiguous description structure, we use the server identifier s_{ij}, the truck identifier u_{nr}, and the transport subsystem identifier h_n as descriptors of e_1-, e_2- and e_3-type events, respectively. An e_4-type event (release of a new production order) can be described by a 15-tuple having the same structure and meaning as the production activity configuration. Finally, the identifier o_γ of the production order to remove and the new scheduling policy to apply are the descriptors of e_5 and e_6 events, respectively.

Now, let E_i $(i = 1, \ldots, 6)$ be the set of the external e_i-type events. The closure of the external event set with the non-event $\varnothing$

$$X^{\varnothing} = \cup_{i=1,6} E_i \cup \{\varnothing\}$$

represents the input set for the DEDS model. Furthermore, for any time interval $[t_0, t_1]$, each function

$$\sigma : [t_0, t_1 \rightarrow X^{\varnothing}$$

is an input segment, i.e., it makes one, and only one, element of $X^{\varnothing}$ correspond to a single instant from $[t_0, t_1]$. The letter Σ denotes the set of all the input segments.

Events from e_7 to e_{10} are internal, because they arise from the system dynamics. Although they do not require any specific description structure, it is necessary to specify their occurrence times exactly. To this end, we have to preliminarily define the system state. So, let $e(t)$ be the time elapsed from the last event occurrence to the current time t. The pair

$$\mathbf{q}(t) = [\mathbf{C}(t), e(t)]$$

© 2001 by CRC Press LLC

represents the (total) DEDS state at time t [20], [21], [22]. So, the values of $\mathbf{q}(t)$, for $t \in R^+$, trace the system trajectory. More clearly, the changes of entities, attributes, and relationships from $\mathbf{F}(t)$ and the modifications of $\Omega(t)$, RT(t), and $e(t)$ formally describe the system behavior. Note that $e(t)$ changes continuously with time as a clock variable and is reset to zero in correspondence of every event occurrence.

Coming back to the characterization of the internal event occurrence times, we introduce three more job subsets J_{Ru}^*, J_C^*, and J_{TW}^*

$$\begin{aligned} J_{Ru}^* &= \{j_k \in J_{Ru} | \text{BA}(j_k) = 0 \quad \text{and} \quad \text{SCI}(j_k) = 0\} \\ J_C^* &= \{j_k \in J_c | \text{BA}(j_k) = 0, \quad \text{SCI}(j_k) = 0, H\text{-STATUS}\{\text{BEL-}U[\text{CAR}(j_k)]\} \\ &= Up \quad \text{and} \quad U\text{-STATUS}[\text{CAR}(j_k)] = Carrying\} \\ J_{TW}^* &= \{j_k \in J_{TW} | \text{SCI}(j_k) = 0\} \end{aligned}$$

More precisely, J_{Ru}^* collects all the nonblocked and noninhibited *Running* jobs, J_C^* contains all the nonblocked and noninhibited jobs *Carried* by *Carrying* transport units of *Up* transport subsystems. Finally, J_{TW}^* groups all noninhibited *Waiting for transport* jobs. An internal event occurs at time t if, and only if, the following condition is true

$$e(t) = \min \{\text{RT}, \text{ROT}(j_k) \; \forall \, j_k \in J_{Ru}^* \cup J_{TW}^* \cup J_C^*\} \tag{1.1}$$

where the occurring event is e_7, if $e(t) = \text{RT}$, or $e_8(e_9, e_{10})$, if $e(t) = \text{ROT}(j_k^*)$ with $j_k^* \in J_{Ru}^* (J_C^*, J_{TW}^*)$.

The DEDS Model

Now, we are in the position to define our DEDS model as the 7-tuple

$$S = \langle \tau, X^{\varnothing}, \Sigma, Q, Y, \delta, \eta \rangle$$

Note that τ is the time set (equal to the positive real set R^+), Q represents the total state set, Y indicates the output set, while the functions

$$\delta: \; Q \times \Sigma \to Q$$

and

$$\eta: Q \to Y$$

are the state transition and the output functions, respectively. In this context, we assume the output coincident with the system's state ($Y = Q$).

To complete the definition of the DEDS model, we must specify the state transition function only. Starting from a state $\mathbf{q}(t_0)$ at time t_0, and applying an input segment σ defined on $[t_0, t_1]$, this function allows us to know the system's state at time t_1

$$\mathbf{q}(t_1) = \delta[\mathbf{q}(t_0), \sigma]$$

If no event occurs in the interval $[t_0, t]$, the pair composing the state $\mathbf{q}(t)$ is

$$\mathbf{C}(t) = \mathbf{C}(t_0) \tag{1.2}$$

$$e(t) = e(t_0) + t - t_0 \tag{1.3}$$

© 2001 by CRC Press LLC

On the other hand, if an event occurs at time t, denoting by $\mathbf{q}(t)$ and $\mathbf{q}(t^+)$ the states just before and just after the event occurrence, respectively, it holds

$$e(t^+) = 0 \tag{1.4}$$

and

$$\mathbf{C}(t^+) \neq \mathbf{C}(t) \tag{1.5}$$

The mechanism driving the sequential state from $\mathbf{C}(t)$ to $\mathbf{C}(t^+)$ can be described by the decision table technique [16]. Decision tables may be thought as a high-level nonalgorithmic language, whose syntax is defined in a straightforward fashion and whose semantic is the mapping. They are suitable to avoid incompleteness and ambiguity and, therefore, to describe the complex mechanisms of the event-driven state transition. The elements of a decision table are (a) conditions, (b) actions, (c) rules in form of an allocation matrix, and (d) action entry. In the DEDS model, the conditions test information on **F** and the actions indicate the corresponding state modifications, using decision rules if some choice is necessary. Due to the limited space of this exposition, however, we do not provide the details of the tables, which are completely reported in [6].

Although an explicit formulation for δ is not feasible for our DEDS model, by the composition property of dynamic systems [20], [21], [22], decision tables, and equations (1.2), (1.3), and (1.4) define the state transition function implicitly. In fact, the state trajectory results from successively considering the state instantaneous updating, triggered by an event occurrence, and the mechanism for the clock variable advancing. Of course, the above composition requires the (external and internal) events be completely time ordered. Such a fundamental condition call for a convention, establishing the order in which one must consider simultaneous events, if any, for generating the corresponding state transitions. As an example, assuming the external events (i.e., the system's input) be completely ordered, we can state the following rules for simultaneous events (a) internal events are considered before external ones, (b) internal events are considered in the following order: e_7, e_8, e_9, and e_{10}, and *(c)* given a rule for ordering jobs in J, internal events of the same type are sequenced as the corresponding jobs.

1.9 An Example of How the Modeling Process

In this section, we apply our formalism to a case study inspired by the General Electric FMS at Erie, Pennsylvania, described by Viswanadham et al. in [17]. This system consists of seven workstations provided with input and output buffers and of an AGV system with only one truck. Figure 1.2 shows the system layout and specifies the role of each workstation. This FMS has to manufacture locomotive parts of two types called 509 and 640. Parts of type 509 are obtained by a working procedure of 17 operation steps and parts of type 640 by a working procedure of 18 operation steps.

To build the FMS model we specify the structural view, the entities, and the relationships of the queuing network configuration and the scheduling policies. While the elements of the structural view are fixed, those characterizing queuing network configuration and scheduling policy depend on the current state of the system. So, for sake of simplicity, we refer to a defined state describing hypothetical conditions of the system.

The Model Static Structure of the Application Example

First of all, we define the following entity sets and the associated relations and attributes

$$M = \{m_i,\ i = 1, 2, \ldots, 7\}$$
$$B = \{b_\mu,\ \mu = 1, 2, \ldots, 14\}$$

© 2001 by CRC Press LLC

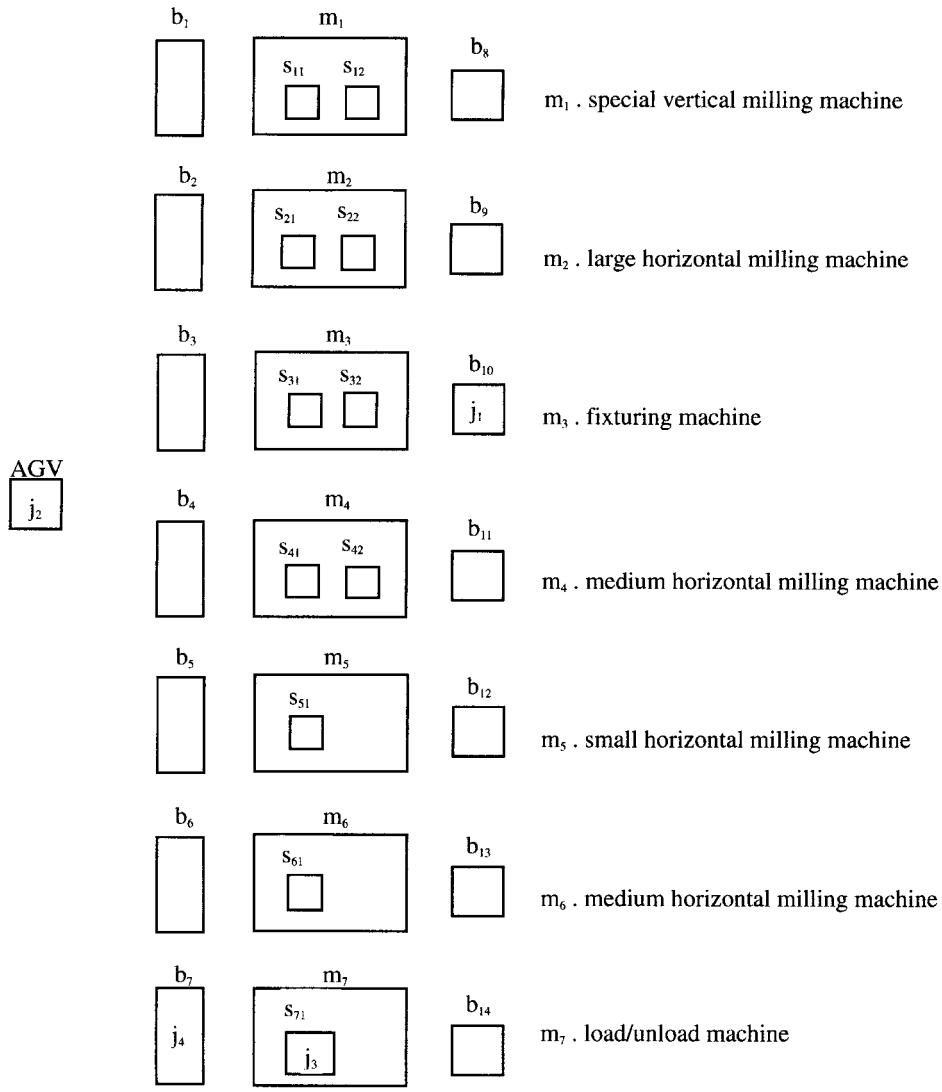


FIGURE 1.2 Lay-out of the production system in the example.

with

$$\begin{aligned} \text{TB}(b_\mu) &= \text{I}, \quad \text{BC}(b_\mu) = 2 \quad \text{for } \mu = 1,\ldots,7 \\ \text{TB}(b_\mu) &= \text{O}, \quad \text{BC}(b_\mu) = 1 \quad \text{for } \mu = 8,\ldots,14 \\ \text{BEL-}B(b_i) &= \text{BEL-}B(b_{i+7}) = m_i \quad \text{for } i = 1,\ldots,7 \end{aligned}$$

Each machine m_i has S_i servers with $S_i = 2$ for $i = 1,\ldots,4$ and $S_i = 1$ for $i = 5, 6, 7$ composing the following set

$$S = \{s_{ij}, i = 1, 2,\ldots, 7, j = 1, 2 \quad \text{for} \quad i = 1,\ldots,4 \quad \text{and} \quad j = 1 \quad \text{for} \quad i = 5, 6, 7\}$$

© 2001 by CRC Press LLC

with

$$\text{BEL-}S(s_{ij}) = m_i \text{ for } i = 1, 2, \ldots, 7, j = 1, 2 \text{ for } i = 1, \ldots, 4 \text{ and } j = 1 \text{ for } i = 5, 6, 7$$

Since the FMS is provided with an AGV system consisting of one truck u_1, we get $H = \{h_1\}$, $U = \{u_1\}$, and BEL-$U(u_1) = h_1$. Assuming all the transportation times equal to two time units (t.u.) for any couple of starting and destination machines, it holds $T = \{(h_1, m_k, m_j) \mid \text{for each } m_k, m_j \in M \text{ with } m_k \neq m_j\}$ and $\text{TT}(h_1, m_k, m_j) = 2$ (t.u.) for each $(h_1, m_k, m_j) \in T$.

The Queuing Network Configuration of the Application Example

Let us assume that the FMS has only one production order o_1 currently in execution, to manufacture two final products p_1 and p_2, representing 509 and 640 part types. Thus actual job type and order entity sets are $P = \{p_1, p_2\}$ and $O = \{o_1\}$ with BEL-$P(p_\pi) = o_1$ for $\pi = 1, 2$.

Due dates and product numbers complete the production order description

$$\text{DD}(o_1) = \text{K t.u.} \qquad \text{PN}(p_\pi) = 100 \qquad \text{for } \pi = 1, 2$$

Each p_π-type final product is obtained by one working procedure, so that

$$W = \{w_\pi, \pi = 1, 2\} \text{ with BEL-}W(w_{\pi'}) = p_\pi, \quad \pi = 1, 2$$

and all the elements from W represent initial working procedures: $W_I = W$. Moreover, since the system does not perform any assembly operation, the set A is empty.

The operation steps composing the working procedures are elements of the set

$$V = \{v_{1,1}, \ldots, v_{1,19}, v_{2,1}, \ldots, v_{2,20}\}$$

with

$$\begin{aligned} \text{BEL-}V(v_{1,\alpha}) &= p_1 \quad \text{for } \alpha = 1, \ldots, 19 \\ \text{BEL-}V(v_{2,\alpha}) &= p_2 \quad \text{for } \alpha = 1, \ldots, 20 \\ \text{TS}(v_{\pi,\alpha}) &= N \quad \text{for each } v_{\pi,\alpha} \in V \end{aligned}$$

In each working procedure, any step is followed by only one operation, but $v_{1,5}$, $v_{1,11}$, $v_{2,6}$, and $v_{2,12}$ that precede two alternative steps. More precisely, the alternative routing relationship is defined by the set

$$\begin{aligned} R = \{&(v_{1,\alpha}, v_{1,\alpha+1}) \text{ for } \alpha = 1, \ldots, 5, \quad (v_{1,5}, v_{1,7})(v_{1,6}, v_{1,8}) \quad (v_{1,\alpha}, v_{1,\alpha+1}) \text{ for } \alpha = 7, \ldots, 11, \\ &(v_{1,11}, v_{1,13})(v_{1,12}, v_{1,14}) \quad (v_{1,\alpha}, v_{1,\alpha+1}) \text{ for } \alpha = 13, \ldots, 18, \\ &(v_{2,\alpha}, v_{2,\alpha+1}) \text{ for } \alpha = 1, \ldots, 6, \quad (v_{2,6}, v_{2,8})(v_{2,7}, v_{2,9}) \, (v_{2,\alpha}, v_{2,\alpha+1}) \text{ for } \alpha = 8, \ldots, 12, \\ &(v_{2,12}, v_{2,14})(v_{2,13}, v_{2,15}) \quad (v_{2,\alpha}, v_{2,\alpha+1}) \text{ for } \alpha = 14, \ldots, 20\} \end{aligned}$$

Since any operation step requires only one machine, $\text{COM}(v_{\pi\alpha})$ is empty for each $v_{\pi,\alpha} \in V$. Moreover, Table 1.1 reports the functions PT and JHM.

© 2001 by CRC Press LLC

TABLE 1.1 Job Hosting Machines and Processing Times for the Operation Steps in the Case Study

w_1	JHM	PT	w_2	JHM	PT
$v_{1,1}$	m_7	20	$v_{2,1}$	m_7	20
$v_{1,2}$	m_3	30	$v_{2,2}$	m_3	30
$v_{1,3}$	m_2	112	$v_{2,3}$	m_1	101
$v_{1,4}$	m_1	168	$v_{2,4}$	m_2	70
$v_{1,5}$	m_2	22	$v_{2,5}$	m_1	59
$v_{1,6}$	m_6	36	$v_{2,6}$	m_2	17
$v_{1,7}$	m_4	36	$v_{2,7}$	m_6	26
$v_{1,8}$	m_3	20	$v_{2,8}$	m_4	26
$v_{1,9}$	m_7	20	$v_{2,9}$	m_3	20
$v_{1,10}$	m_2	78	$v_{2,10}$	m_7	20
$v_{1,11}$	m_5	93	$v_{2,11}$	m_2	64
$v_{1,12}$	m_6	42	$v_{2,12}$	m_5	10
$v_{1,13}$	m_4	42	$v_{2,13}$	m_6	52
$v_{1,14}$	m_7	20	$v_{2,14}$	m_4	52
$v_{1,15}$	m_3	15	$v_{2,15}$	m_7	20
$v_{1,16}$	m_4	69	$v_{2,16}$	m_3	15
$v_{1,17}$	m_2	13	$v_{2,17}$	m_6	56
$v_{1,18}$	m_3	15	$v_{2,18}$	m_2	16
$v_{1,19}$	m_7	20	$v_{2,19}$	m_3	15
			$v_{2,20}$	m_7	20

We suppose that in the current state there are $N_J = 4$ jobs in the system: $J = \{j_1, j_2, j_3, j_4\}$. Table 1.2 lists all the job attributes.

TABLE 1.2 Job Attributes in the Current State

	Jobs			
Attributes	j_1	j_2	j_3	j_4
JT *J*-STATUS	p_1 *Waiting for destination*	p_2 *Carried*	p_1 *Running*	p_2 *Ready*
ROS	$v_{1,2}$	$v_{2,2}$	$v_{1,1}$	$v_{2,1}$
LOS	$v_{1,1}$	$v_{2,1}$	n.o.s.	n.o.s.
HOS	b_{10}	–	–	b_7
CAR	–	u_1	–	–
ROT	∞	2 t.u.	20 t.u.	20 t.u.
BA	0	0	–	–
SCI	0	0	0	0
SAT	0	20 t.u.	22 t.u.	40 t.u.
BAT	52 t.u.	–	–	40 t.u.

Finally, according to the job configuration, only s_{71} is processing a job in the current state. On the other hand a job is holding the AGV truck. In particular we get

$$
\begin{aligned}
S\text{-STATUS}(s_{ij}) &= \textit{Idle} \text{ for } i = 1, 2, \ldots, 6,\ j = 1, 2 \text{ for } i = 1, \ldots, 4 \text{ and } j = 1 \text{ for } i = 5, 6 \\
S\text{-STATUS}(s_{71}) &= \textit{Running} \\
\text{PRO}(s_{71}) &= j_3 \\
U\text{-STATUS}(u_1) &= \textit{Carrying} \\
H\text{-STATUS}\ (h_1) &= \textit{Up}
\end{aligned}
$$

© 2001 by CRC Press LLC

The Applied Scheduling Policy

The current scheduling rules complete the state description. To illustrate an important case, we consider a scheduling policy for the deadlock avoidance at the job release and the job flow control level. Deadlocks, indeed, are highly unfavorable conditions in which parts in a set request the resources held by other parts in the same set [11]. These situations are difficult to predict and can propagate to the entire system, blocking the production indefinitely. The scheduling rules here illustrated apply the deadlock avoidance strategies proposed by Fanti et al. [8], [9], [10], [11] and called restriction policies.

To establish the restriction policies, we exploit a digraph-based characterization of the current interactions between jobs and resources, by introducing the transition digraph $D_{Tr}(\mathbf{q}) = [N_{Tr}, E_{Tr}(\mathbf{q})]$ defined for each system state $\mathbf{q}$. In particular, the set of nodes N_{Tr} corresponds to the set of resources $M \cup B \cup H \cup \{r_R\}$, where r_R is a fictitious resource representing the output of the system. Each resource (but r_R) has a finite capacity equal to $BC(b_\mu)$, S_i, or N_n, in case the resource at issue represents a buffer, a machine, or a transport subsystem, respectively. On the contrary, r_R is considered of infinite capacity. Moreover an edge $e_{im} \in E_{Tr}(\mathbf{q})$ iff a *j*ob $j \in J$ holds $r_i \in M \cup B \cup H$ in the state $\mathbf{q}$ and requires $r_m \in M \cup B \cup H \cup \{r_R\}$ as next resource.

As shown in [8], deadlock conditions are related to some particular strong components of the digraph $D_{Tr}(\mathbf{q})$ called maximal-weight zero-outdegree strong components (MZSC). An MZSC has the following properties (a) all the resources from the strong component are busy (Maximal-weight), and (b) all the edges starting from vertices of the strong component end at nodes of the same component (zero-outdegree). As proved in [8], $\mathbf{q}$ is a deadlock state iff there exists at least one MZSC in $D_{Tr}(\mathbf{q})$. The control action for avoiding deadlock consists in inhibiting or enabling job loading and the job status changes to prevent the transition digraph from containing any MZSC [9], [10]. So, applying functions ω_{ls}, ω_{sci}, and ω_i requires the analysis of the transition digraph that can be easily updated event by event. In the following we show how to define decision rules in this case.

Job loading decision rules. For the job type sequencing, we assume a periodic order as the following $(p_1, p_2, p_1, \ldots)$. So, at each event occurrence, a job enters the system if the first resource of its working procedure has at least an idle place in the input buffer and if the transition digraph $D_{Tr}(\mathbf{q}')$ does not contain any MZSC. Here $\mathbf{q}'$ denotes the state the system should reach from $\mathbf{q}$ as a consequence of the job loading.

We define the sequencing and timing functions in two steps.

Step 1: choice of the job type
Let j_{k1} be the last job loaded into the system, i.e., $\mathrm{SAT}(j_{k1}) = \max\{\mathrm{SAT}(j_k)\ \forall\ j_k \in J\}$. Then the next job type to load into the system is p_π, with $p_\pi \neq \mathrm{TJ}(j_{k1})$.

Step 2: specification of the job loading decision rules
If the $\mathrm{JHM}(v_{\pi,1})$ has an idle place in the buffer and if $D_{Tr}(\mathbf{q}')$ does not contain any MZSC, then $\omega_{ls}(\mathbf{F}) = \mathrm{w}_\pi$ and $\omega_{lt}(\mathbf{F}) = 0$ else

$$\omega_{ls}(\mathbf{F}) = \{\text{n.c.}\} \qquad \text{and} \qquad \omega_{lt}(\mathbf{F}) = \{\infty\}.$$

State change inhibition rule. Let $\mathbf{q}'$ be the state the system should reach from $\mathbf{q}$ as a consequence of the job changing status. Then $\omega_{sci}(\mathbf{F}, j_k) = 1$ only if $D_{Tr}(\mathbf{q}')$ contains an MZSC including the resource held by j_k.

Job routing decision rules.

1. Decision function ω_v
   For a given $v_{\pi\alpha}$, if the set V has only one element $v_{\pi B}$ such that $(v_{\pi\alpha}, v_{\pi\beta}) \in R$ then $\omega_v(\mathbf{F}, v_{\pi\alpha}) = v_{\pi\beta}$
   If V contains more couples with the same first element (i.e., $v_{\pi\alpha}$), then $\omega_v(\mathbf{F}, v_{\pi\alpha})$ is chosen such that $\mathrm{JHM}(\omega_v(\mathbf{F}, v_{\pi\alpha}))$ is the machine with the highest number of idle servers. If many workstations have the same number of idle servers, the machine with the highest number of free places in the input buffer is considered. If the ambiguity still remains, ω_v selects the next step with minimum index β.

© 2001 by CRC Press LLC

2. Decision function ω_{v0}

$$\text{If } w_\pi = w_1 \quad \text{then} \quad \omega_{v0}(\mathbf{F}, w_\pi) = v_{1,1} \quad \text{else} \quad \omega_{v0}(\mathbf{F}, w_\pi) = v_{2,1}$$

Priority decision rules.

1–2. Decision functions ω_j and ω_{jh}
Among pieces waiting for service from a machine (or waiting for destination) we select the job arrived first at the input (output) buffer, i.e., the one with minimum BAT attribute.

3–4. Since the system does not perform any assembly operation, there is no need to define $\omega_{a,R}(\mathbf{F}, j_k)$ and $\omega_{a,RA}(\mathbf{F}, j_k)$.

5. Decision function ω_i
Among the inhibited pieces that are currently allowed to modify their status, ω_i selects the one with minimum SAT. More precisely, let us denote with J_i the subset of J whose generic element (say j_k) enjoys the following properties: $\text{SCI}(j_k) = 1$ and $D_{Tr}(\mathbf{q}')$ does not contain any MZSC ($\mathbf{q}'$ stands for the state the system should reach from $\mathbf{q}$ on j_k-status changing). So, $\omega_i(\mathbf{F})$ is the job from J_i with minimum SAT attribute. If J_i is empty then $\omega_i(\mathbf{F}) = \text{n.c.}$

6–8. Decision functions ω_s, ω_{fs}, ω_{bs}
The selection of an *Idle* server of a machine m_i (decision function ω_s) as well as the selection of a server to unblock next ($\omega_{\bar{fs}}$) are conventionally performed by choosing the *Idle* (*Blocked*) server s_{ij} with minimum index j. Moreover, as no operation step requires cooperating machines, the selection of the first server to consider available for a new operation (ω_{fs}) can be defined arbitrarily. In fact, in this case the set $\underline{S}$ for which $\omega_{fs}(\mathbf{F}, \underline{S})$ is evaluated always contains only one server.

9–12. Decision functions ω_h, ω_{bh}
Since U contains one element only, i.e., u_1, we get

$$\begin{aligned} \omega_h(\mathbf{F}, m_i, m_j) &= u_1 \quad \text{if } U\text{-STATUS}(u_1) = \mathit{Idle} \\ \omega_h(\mathbf{F}, m_i, m_j) &= \text{n.c.} \quad \text{otherwise} \end{aligned}$$

and

$$\begin{aligned} \omega_{bh}(\mathbf{F}, m_i) &= u_1 \quad \text{if } U\text{-STATUS}(u_1) = \mathit{Blocked} \\ \omega_{bh}(\mathbf{F}, m_i) &= \text{n.c.} \quad \text{otherwise} \end{aligned}$$

As we remark in Section 1.8, the decision table technique is an appropriate mechanism to update the system state on any event occurrence. For sake of brevity it is impossible to report here all the decision tables required by each event. So, as an example, Table 1.3 shows only the decision table concerning the event transport operation completion (e_9). In correspondence of actions A_3, A_4, and A_6 such table calls three more support tables named *A new piece reaches a machine, Selection of the next job to transport,* and *Resetting the State Change Inhibitor*, respectively. Table 1.4 shows only the first one of these, i.e., *A new piece reaches a machine*. Decision tables utilize the auxiliary function

$$\text{FREE}: B \rightarrow \text{I}^+$$

where $\text{FREE}(b_\mu)$ indicates the number of idle places in the buffer b_μ.

1.10 The Generic Control Software Organization

In this section we introduce a general architecture identifying the main components of the controller and the way in which such components interact by sending messages to one another and to the environment. To define the control software architecture, we apply the following criteria [1]:

© 2001 by CRC Press LLC

TABLE 1.3 Decision Table: *Transport Operation Completion* Event Occurrence Condition: $e(t) = ROT(j_k)$ with $j_k \in J_C^*$

	Rules		
Conditions	R_1	R_2	R_3
C_1	Y	N	N
C_2	/	Y	N
Actions			
A_1	•		
A_2		•	
A_3		•	
A_4		•	
A_5			•
A_6	•	•	•

Conditions:

C_1	$\omega_{sci}(\mathbf{F}, j_k) = 1$
C_2	Putting $b_\mu \in B \mid BT(b_\mu) \in \{I, IO\}$ and $BEL\text{-}B(b_\mu) = JHM[ROS(j_k)]$: $FREE(b_\mu) > 0$.

Actions:

A_1	Set: $SCI(j_k) = 1$
A_2	Set: $U\text{-}STATUS[CAR(j_k)] = Idle$ $RCM[CAR(j_k)] = JHM[ROS(j_k)]$
A_3	Consult the Support Table *A new piece reaches a machine* with input parameter: j_k. Finally, return to this table for executing the remaining actions in the rule (if any).
A_4	Consult the Support Table *Selection of the next job to transport*.
A_5	Set: $BA(j_k) = 1$ $U\text{-}STATUS[CAR(j_k)] = Blocked$
A_6	Consult the Support Table *Resetting the State Change Inhibitor*

1. The software must possess a modular organization, where each module is easily integrated into a whole. Modularity is, indeed, a key feature when the reuse of existing software (e.g., planning software, software managing hardware at the SF level, following-up software, data base management system, etc.) is a valuable objective.
2. The role of the modules as well as the interfaces between them must remain unchanged, even if the configuration changes from case to case to fit different factory needs or to execute more and more functions. This role invariance confers the demanded genericity to the control software.

The object-oriented approach to the design of the FMS control software is effective in fitting these criteria. The modeling emphasis is on the patterns, i.e., both on the modules (objects) and on the architecture integrating such objects into a whole [1], [2]. The objects account for the needs the software must satisfy (need patterns), while the architecture expresses the relationships between any two objects and their linkages with other objects external to the software (design patterns). According to [3] and [4], we describe the FMS control software both in terms of a hierarchy of control levels and in terms of content of the component objects.

Then in the following we specify [5]:

- The location of the FMS controller within the hierarchy of the control levels.
- The FMS knowledge the controller includes and the communication it maintains with adjacent levels.
- The objects making up the control software, the activities they execute, the piece of knowledge they contain, and the messages they exchange each other.

© 2001 by CRC Press LLC

TABLE 1.4 Support Table: *A new piece reaches a machine* Input parameter: j_k

	Rules				
Conditions	R_1	R_2	R_3	R_4	R_5
C_3	Y	Y	N	N	N
C_4	/	/	N	Y	Y
C_5	N	Y	/	N	Y
Actions					
A_7	•				
A_8	•			•	
A_9		•			
A_{10}		•			•
A_{11}			•	•	
A_{12}				•	
A_{13}					•
A_{14}					•

Let us indicate $TJ(j_k) = \pi$ and $b_\mu \in B \mid BT(b_\mu) \in \{I, IO\}$ and $JHM[ROS(j_k)] = BEL\text{-}B(b_\mu)$

Conditions:

C_3 $TS[ROS(j_k)] = N$

C_4 Putting $w_{\pi\beta} = BEL\text{-}P[ROS(j_k)]$, $w_{\pi\alpha} = BEL\text{-}P[LOS(j_k)]$, {$\exists$ WEI $[(w_{\pi\alpha}, w_{\pi\beta})] - 1$ jobs $j_h \mid LOS(j_h) = LOS(j_k)$ and $ROS(j_h) = ROS\ (j_k)$ and *J*-STATUS(j_h) = *Ready for assembling* and $SCI(j_h) = 0$}
and
{for each $w_{\pi\delta} \in W \mid w_{\pi\delta} = w_{\pi\alpha}$ and $(w_{\pi\delta}, w_{\pi\beta}) \in A$ $\exists$ WEI $[(w_{\pi\delta}, w_{\pi\beta})]$ jobs $j_h \mid BEL\text{-}P[LOS(j_h)] = w_{\pi\delta}$ and $ROS(j_h) = ROS(j_k)$ and *J*-STATUS(j_h) = *Ready for assembling* and $SCI(j_h) = 0$}

C_5 for each $m_i \in \{JHM[ROS(j_k)]\} \cup COM[ROS(j_k)]$ $\exists\, s_{ij} \mid BEL\text{-}S(s_{ij}) = m_i$ and *S*-STATUS(s_{ij}) = *Idle*

Actions:

A_7 Set: *J*-STATUS(j_k) = *Ready*, $FREE(b_\mu) = FREE(b_\mu) - 1$
$HOS(j_k) = b_\mu$, $ATB(j_k) = t$, $ROT(j_k) = PT[ROS(j_k)]$

A_8 for each $m_i \in \{JHM[ROS(jk)]\} \cup COM[ROS(jk)]$ $\exists\, s_{ij} \mid BEL\text{-}S(s_{ij}) = m_i$ and *S*-STATUS(s_{ij}) = *Idle*
Set: *S*-STATUS$[\omega_s(\mathbf{F}, m_i)]$ = *Reserved*
$RES[\omega_s(\mathbf{F}, m_i)] = j_k$

A_9 Set: *J*-STATUS(j_k) = *Running*
$ROT(j_k) = PT[ROS(j_k)]$
$BAT(j_k) = t$

A_{10} for each $m_i \in \{JHM[ROS(j_k)]\} \cup COM[ROS(j_k)]$
Set: S-STATUS$[\omega_s(\mathbf{F}, m_i)]$ = *Running*
$PRO[\omega_s(\mathbf{F}, m_i)] = j_k$

A_{11} Set: *J*-STATUS(J_k) = *Ready for assembling*
$FREE(b_\mu) = FREE(b_\mu) - 1$
$HOS(j_k) = b\mu$
$ATB(j_k) = t$
$ROT(j_k) = PT[ROS(j_k)]$

A_{12} Set: *J*-STATUS(j_k) = *Ready*
for each $j_h \in \omega_{a,RA}(\mathbf{F}, j_k)$
Set: *J*-STATUS(j_h) = *Ready*

A_{13} For each $j_h \in \omega_{a,RA}(\mathbf{F}, j_k)$
Set: *J*-STATUS(j_h) = *Ready*

A_{14} Set: *J*-STATUS(j_k) = *Running*
$ROT(j_k) = PT[ROS(j_k)]$
$ATB(j_k) = t$
For each $j_h \in \omega_{a,RA}(\mathbf{F}, j_k)$
Set: $J = J - \{j_h\}$
FREE $(b_\mu) = FREE(b_\mu) + 1$

© 2001 by CRC Press LLC

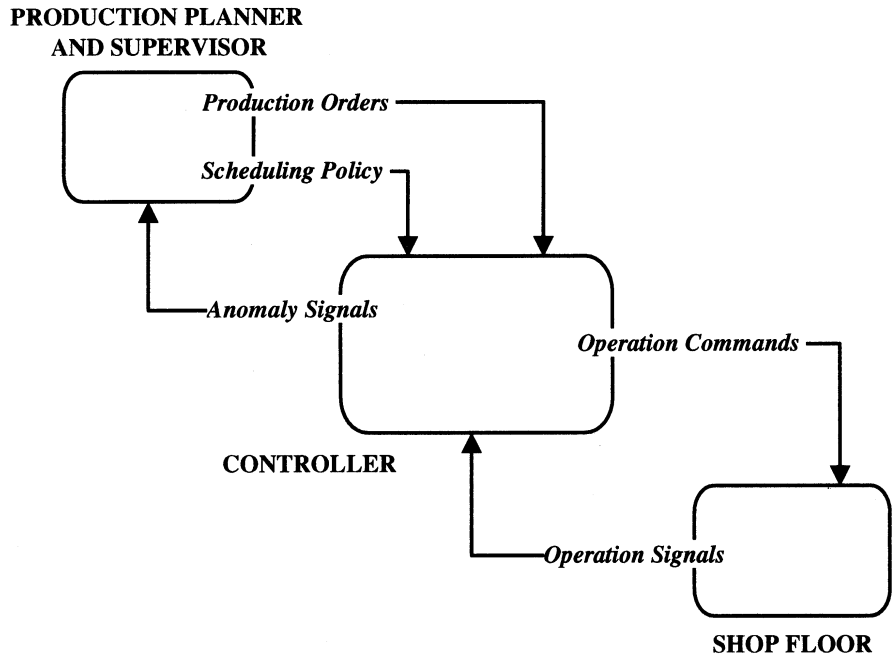


FIGURE 1.3 Hierarchy of the control levels.

The Control Hierarchy

The control software is modular and contains the information necessary to transform the production orders released by the higher decision level (production planning) into commands addressed to the lower level (shop floor level). These commands trigger the operation sequence which satisfies the order dispatched at the production planning level. Moreover the control software notifies the production planning level of operation anomalies occurred at SF level.

For the purpose of this paper, a single module called production planner and supervisor represents the activities at the production planning level, as well as the single module shop floor describes the activities at the SF level (see Figure 1.3). The production planner and supervisor transfers the commands specifying the decision rules of the current scheduling policy and the production orders to the control software. By applying the state transition rules, the control software runs the procedures for updating the queuing network configuration and the residual time to the next job input. Moreover it sends operation commands to SF specifying the tasks to execute.

Objects and Architecture of the Control Software

The DEDS model constitutes the kernel of the software driving the FMS. Namely we use a model-reference scheme to control the SF, by allowing our framework to embody the desired manufacturing features the FMS has to follow. In other words, we require that, when subject to the same external events e.g., orders insertion/removal, scheduling policy updating, etc., the SF behaves in the same way as the reference DEDS model. However we have to overcome some difficulties to implement this model-reference. Due to the uncertainties affecting the values of the processing and transport times, applying condition (1) could lead to determine imprecisely the occurrence times for events e_7, e_8, e_9, and e_{10}. Consequently, the control software could foresee event occurrences different from the sequence of the real production process. To overcome this difficulty, the SF must directly detect and transmit completion of operation

© 2001 by CRC Press LLC

steps, transport operations, and transport unit positioning to the control software in charge of triggering the corresponding state transition. In this way equation (1.1) gives only the expected times of internal event occurrences. Moreover the control software can diagnose possible anomalous conditions of the SF, by monitoring and by comparing the expected and the real completion times.

The information flow from the SF to the control software consists of the detection of e_1-, e_2-, e_3-, e_7-, e_8-, e_9-, and e_{10}-type events whereas no state component is fed back from the SF, as the DEDS state holds the necessary information. Then, the control software must update the data base according to the transition rules triggered by the above event-types and must transmit appropriate signals to the specific control devices at the SF level. Hence, to be capable of following the signals from SF, the control software must guarantee that the mean time actually taken for serving a signal (communication time and information processing time) be shorter than the mean time between two consecutive events. Usually the manufacturing systems possess this capability so that they do not encompass hard real time constraints. Finally the e_4-, e_5-, and e_6-type events only require a DEDS state updating and do not imply immediate actions on the SF.

Figure 1.4 indicates schematically the objects composing the control software and the architecture of such objects. In particular, they are the data base, the communication manager, the job release manager, the job flow manager, the time supervisor, and the diagnostic message handler. The figure describes the knowledge contained in each object and the messages that the objects exchange. Continuous lines indicate synchronous messages while dotted lines show asynchronous messages or signals.

The data base of the FMS encapsulates the current queuing network configuration ($\mathbf{F}(t)$) and the current residual time to the next job input (RT(t)). The data base object possesses the methods for building, using, and updating the relational data base.

The communication manager rules all the messages addressed to the higher and lower level of the control hierarchy. This object holds the queue of the external and internal events and owns the methods for:

- updating and ordering the queue when the production planner and supervisor sends a command, when the SF sends a signal, when the job flow manager forwards job flow decision messages, and when the time supervisor sends a new job loading order; and
- sending to the job release manager the messages involving order insertions/removals (e_4-, and e_5-type events) and sending to the job flow manager the messages about failure and repair occurrences for servers, for AGV trucks and for transport subsystems (e_1-, e_2-, and e_3-type events) and about scheduling policy updating (e_6-type events), next job loading (e_7-type event), completion of operation steps, of transport operations, and of transport unit positioning (e_8-, e_9-, and e_{10}-type events).

The job release manager updates the jobs to be produced according to the orders coming from the production planner and supervisor (e_4- and e_5-type events). It is an object without state. However it contains the methods (decision tables) for:

- calling the data base to retrieve data on the current queuing network configuration and updating the sequential state stored in the data base; and
- transmitting to the job flow manager a command to reset the value of RT(t).

The job flow manager object encapsulates the knowledge about the active rules for job sequencing, job timing, job routing, and priority setting. It owns the methods (decision tables) for dealing with events concerning hardware failures/repairs at the SF level (e_1-, e_2-, and e_3-type events), and implying the change of one or more decision rules within the scheduling policy (e_6-type events), the load of the next job into the system (e_7-type events), the completion of single operations at the SF level (e_8-, e_9-, and e_{10}-type events). The methods of the job flow manager provide for:

- calling the data base of the FMS to retrieve data on the current queuing network configuration and updating the state information stored in the system's data base;
- determining the new value of RT(t);

© 2001 by CRC Press LLC

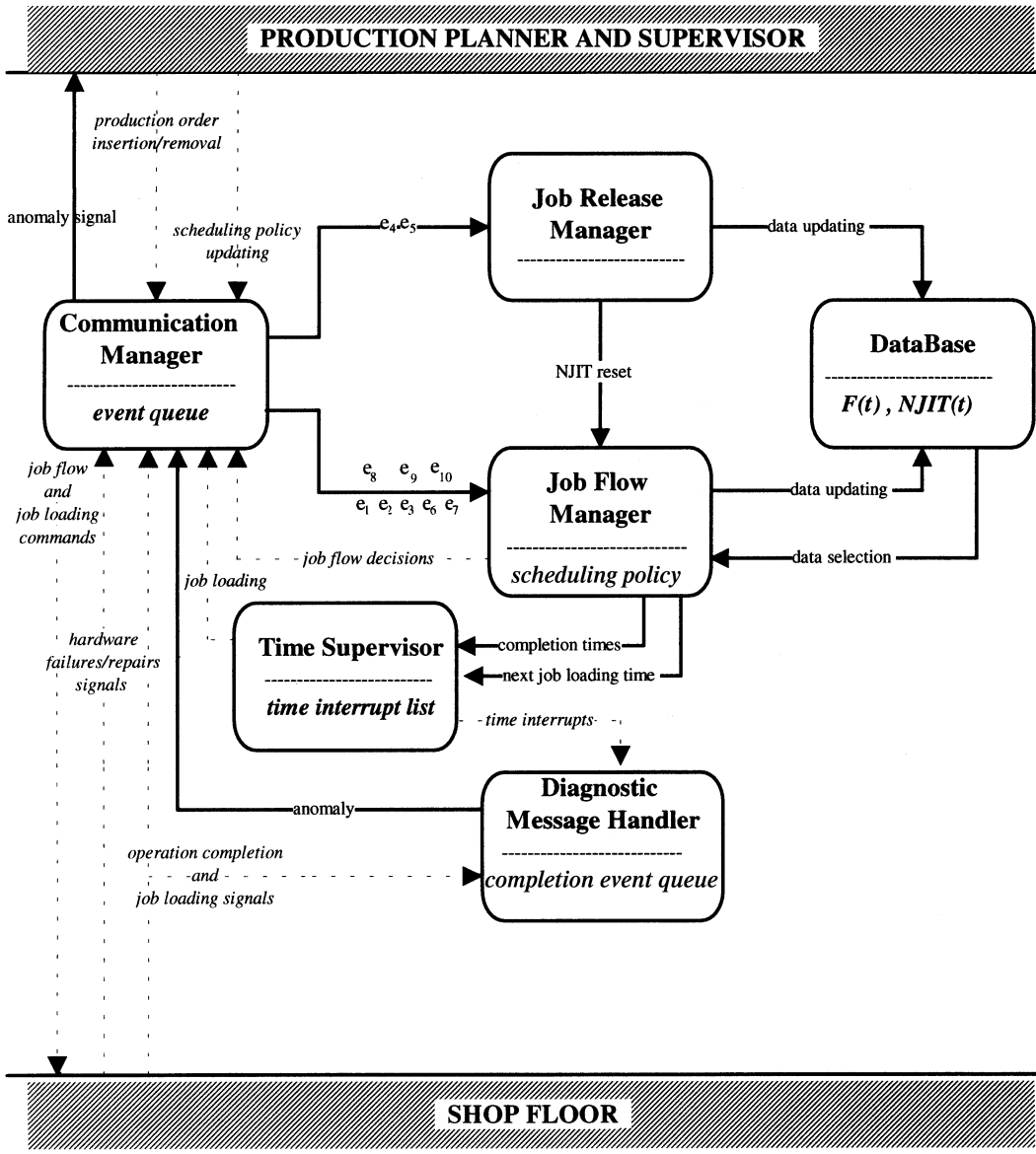


FIGURE 1.4 Objects composing the Control Software.

- transmitting to the communication manager the decisions concerning job flow; and
- sending to the time supervisor the foreseen completion times for operations required to the SF and the new value of RT(t).

The time supervisor guarantees that the foreseen completion times of tasks are signaled. This object holds the list of the foreseen times for the events concerning operation completions at the SF level. It contains the methods for:

- updating and ordering the list because of the arrival of a time setting coming from the job flow manager;
- generating and transmitting to the diagnostic message handler the message signaling the expiration of the time necessary for completing the first operation in the list; and
- signaling the order for queuing an e_7-type event to the communication manager.

© 2001 by CRC Press LLC

The diagnostic message handler is in charge of diagnosing possible anomalous operating conditions of the SF. It encapsulates the list of the internal events currently monitored, each one containing both the expected and real completion times. Moreover, it owns the methods for:

- receiving the messages forwarding the times necessary for completing operations executed at the SF level;
- receiving the messages from the SF and signaling the real completion of an operation;
- signaling the anomaly to the communication manager in case the difference between the real and the expected time for an internal event occurrence exceeds an assigned time-out.

1.11 Conclusions

Automation in manufacturing presents challenging problems to the controls community. This contribution describes our efforts to apply system approach to the development of generic control software for FMS. Namely, our main idea of using a reference DEDS model to drive the SF calls for a system-theoretic abstract formalism describing the dynamics of FMSs. The driving force of this idea is primarily the modeling and abstraction ability offered by the system approach. The proposed framework is general enough for capturing many details of the FMS operation even if some limitations are still present. For example, the tool management is not included in the model or different types of transport systems are not described. However, these restrictions can be removed by enriching the model with further entities, attributes, and relationships. Finally, the object-oriented design of the control software leads to a modular architecture.

Appendix: List of Principal Symbols

Sets and Relationships

B	Buffer Entity Set
J	Job Entity Set
J_C, J_{DW}, J_R, J_{RA}, J_{Ru}, J_{TW}	Subsets of J
H	Transport Subsystem Entity Set
M	Machine Entity Set
O	Order Entity Set
P	Job Type Entity Set
S	Server Entity Set
S_{RuB}, S_R	Subsets of S
T	Transport Entity Set
U	Transport Unit Entity Set
U_{Id}	Subsets of U
V	Operation Step Entity Set
W	Working Procedure Entity Set
W_I	Subsets of W
A	Assembly Relationship
R	Alternative Routing Relationship
RT	Residual Time to the Next Job Input
N_B	Cardinality of B
N_J	Cardinality of J
N_M	Cardinality of M
N_P	Cardinality of P
N_H	Cardinality of H
N_O	Cardinality of O

© 2001 by CRC Press LLC

Attributes		
BA	$J_{Ru} \cup J_C \rightarrow \{0, 1\}$	Blocking Attribute
BAT	$J_R \cup J_{RA} \cup J_{DW} \cup J_{TW} \rightarrow \mathrm{R}^+$	Arrival Time to the Buffer
BC	$B \rightarrow \mathrm{I}^+$	Buffer Capacity
DD	$O \rightarrow \mathrm{R}^+$	Due Date
PN	$O \rightarrow \mathrm{I}^+$	Product Number
PT	$V \rightarrow \mathrm{R}^+$	Processing Time
ROT	$J \rightarrow \mathrm{R}^+ \cup \{\infty\}$	Residual Operation Time
SAT	$J \rightarrow \mathrm{R}^+$	Arrival Time to the System
SCI	$J \rightarrow \{0, 1\}$	State Change Inhibitor
TB	$B \rightarrow \{\mathrm{I}, \mathrm{O}, \mathrm{IO}\}$	Type of Buffer
TJ	$J \rightarrow \mathrm{P}$	Type of Job
TS	$V \rightarrow \{\mathrm{N}, \mathrm{A}\}$	Type of Step
TT	$T \rightarrow \mathrm{R}^+$	Transport Time
WE	$A \rightarrow \mathrm{I}^+$	(Weight) Number of Jobs Concurring in the Assembling
Functions		
BEL-X	$X \rightarrow Y$	Specifies that an element x_i from X belongs to an element y_i from Y
BOO	$J_{TW} \rightarrow U$	Specifies the transport unit booked by a job j_k
CAR	$J_C \rightarrow U$	Specifies the transport unit carrying the job
COM	$V \rightarrow \{\underline{M}\}$	Gives the Cooperating Machines
HOS	$J_R \cup J_{RA} \cup J_{DW} \cup J_{TW} \rightarrow B$	Gives the buffer hosting the job
JHM	$V \rightarrow M$	Gives the Job Hosting Machine
LOS	$J \rightarrow V \cup \{\mathrm{n.o.s}\}$	Defines the last operation step executed by a job
PRO	$S_{RuB} \rightarrow J_{Ru}$	Identifies *Running* or *Blocked* servers in relation with *Running* jobs
RCM	$U_{Id} \rightarrow M$	The transport unit rests close to a machine
RES	$S_R \rightarrow J_R \cup J_{RA}$	Identifies reserved servers
ROS	$J \rightarrow V$	Yields the Reference Operation Step

Acknowledgments

Some text included in this chapter is reprinted with permission from Fanti, M.P., Maione, B., Piscitelli, G., and Turchiano, B., "System Approach to Design Generic Software for Real-Time Control of Flexible Manufacturing System," *IEEE Trans. on Systems, Man and Cybernetics* — Part A, Vol. 26, No. 2, March 1996. ©1996 IEEE.

References

[1] Alexander, C., 1979, *The Timeless Way of Building,* Oxford University Press, New York.

[2] Chen, P. P. 1976, The Entity-Relationship Model—toward a unified view of data, *ACM Transactions on Database Systems,* vol. 1, no. 1, pp. 9–36.

[3] Coad, P. 1992, Object oriented patterns, *Communication of ACM.*

[4] Elia, G., and Menga, G. 1994, Object-oriented design of flexible manufacturing systems, in *Computer Control of Flexible Manufacturing Systems—Research & Development,* S.B. Joshi and J.S. Smith, Eds. pp. 315–342, Chapman & Hall, Boca Raton, FL.

[5] Fanti, M.P., Maione, B., Piscitelli, G., and Turchiano, B., 1992, System approach to a generic software specification for flexible manufacturing system job-flow management, *Int. J. Systems Sci.,* vol. 23, no. 11, pp. 1889–1902.

© 2001 by CRC Press LLC

[6] Fanti, M.P., Maione, B., Piscitelli, G., and Turchiano, B., 1993, A generic software for FMS real-time control, Part I – Part II, DEE Report, no. 3-4/93/S, Politecnico di Bari.

[7] Fanti, M.P., Maione, B., Piscitelli, G., and Turchiano, B., 1996, System approach to design generic software for real-time control of flexible manufacturing systems, *IEEE Trans. Systems, Man and Cybernetics — Part A: Systems and Humans,* vol. 26, no. 2, pp. 190–202.

[8] Fanti, M.P., Maione, G., and Turchiano, B., 1996, Digraph-theoretic approach for deadlock detection and recovery in flexible production systems, *Studies in Informatics and Control,* vol. 5, no. 4, pp. 373–383.

[9] Fanti, M.P., Maione, B., and Turchiano, B., 1996, Graph-theoretic methodology to define deadlock avoidance policies, *Proc. IEEE-SMC Symposium on Discrete Events and Manufacturing Systems, CESA'96,* pp. 419–425, Lille, France.

[10] Fanti, M.P., Maione, B., and Turchiano, B., 1996, Deadlock avoidance policies for flexible production systems with multiple capacity resources, *Proc. IEE 3rd Workshop on Discrete Event System,* pp. 31–38, Edinburgh, Scotland, UK.

[11] Fanti, M.P., Maione, B., Mascolo, S., and Turchiano, B., 1997, Event-based feedback control for deadlock avoidance in flexible production, *IEEE Trans. on Robotics and Automation*.

[12] Huguet, P. and Grabot, B., 1995, Object oriented analysis and design reference models for production activity control system development, Proceedings 1995 INRIA/IEEE Symposium on Emerging Technologies and Factory Automation, ETFA'95 (10–13 October), pp. 413–420, Paris.

[13] Martin, J., and McClure, C., 1985, *Diagramming Techniques for Analysts and Programmers,* Prentice-Hall, Englewood Cliffs, NJ.

[14] Naylor, A.W., and Maletz, M.C., 1986, The manufacturing game: a formal approach to manufacturing software, *IEEE Trans. on Systems, Man, and Cybernetics,* vol. 16, no. 3, pp. 321–334.

[15] Naylor, A.W., and Volts, R. A., 1987, Design of integrated manufacturing system control software, *IEEE Trans. on Systems, Man, and Cybernetics,* vol. 17, no. 6, pp. 881–897.

[16] Pollak, S.L., Hicks, H. T. Jr., and Harrison, W.J., 1971, *Decision Tables: Theory and Practice,* John Wiley & Sons, New York.

[17] Viswanadham, N., Narahari, Y., and Johnson, T.L., 1990, Deadlock prevention and deadlock avoidance in flexible manufacturing systems using petri nets models, *IEEE Trans. on Robotics and Automation,* vol. 6, no. 6, pp. 713–723.

[18] Wymore, A.W., 1967, *A Mathematical Theory of System Engineering. The Elements*, John Wiley & Sons, New York.

[19] Zadeh, L. A., and Desoer, C. A., 1963, *Linear System Theory. The State Space Approach,* McGraw-Hill, New York.

[20] Zeigler, B.P., 1976, *Theory of Modelling and Simulation,* Krieger, Malabar, FL.

[21] Zeigler, B.P., 1984, *Multifacetted Modelling and Discrete Event Simulation,* Academic Press, London.

[22] Zeigler, B.P., 1990, *Object-Oriented Simulation with Hierarchical, Modular Models,* Academic Press, San Diego.

© 2001 by CRC Press LLC

2

Computer-Integrated Assembly for Cost Effective Developments

Rinaldo C. Michelini
University of Genova

Gabriella M. Acaccia
University of Genova

Massimo Callegari
University of Genova

Rezia M. Molfino
University of Genova

Roberto P. Razzoli
University of Genova

2.1 Introduction
2.2 Assembly in Intelligent Manufacturing
Market-Driven Trends in Factory Automation • Cost Effectiveness by Means of Flexibility • The Technology of the Assembly Process
2.3 Effectiveness Through Flexibility Issues
Assessment of the Flexibility Requirements • Decision Supports and Simulation • Example Developments
2.4 Reconfigurable Set-Ups Assembly Facilities
Modular Assembly Transfer Lines • Modularity of Assembly Lines with Buffers and By-Passes
2.5 High-Versatility Self-Fixturing Facilities
Robot-Operated Assembling Set-Ups • Assembling by Integrated Control-and-Management
2.6 Concluding Comments
• Off-Process Setting of a Customer-Driven Mass Production Assembly Facility • Exploiting Recovery Flexibility with Adaptive Modular Assembly Facilities • Programming Optimal Assembly for One-of-a-Kind Products •
2.7 Acknowledgments

2.1 Introduction

For many manufacturing enterprises, assembly is an important portion of the final costs. Effectiveness was traditionally hunted for by reducing complex schedules into unit tasks (scientific work organization) and by enabling sequential assembly lines (vertical flow-shop). The approach leads to the highest productivity, and it is prised for mass production. Flow-lines and fixed schedules, however, require amortisation plans based on steady programmes on duty horizons corresponding to product volumes exceeding some minimal threshold. Market saturation and trade instability look for quick updating of the offered items, properly adapted to wider classes of buyers' needs, possibly, down to the limit situation of one-of-a-kind customised quality. Worldwide enterprises looking for purchasers' oriented supply are, thus, concerned by time-varying artefacts; extended mixes of items have to be processed in parallel and delivered with short time-to-market.

The emphasis toward customised quality, product variety, frequently up-dated offers, quick delivery dates, etc. needs a new approach to effectiveness, exploiting knowledge-intensive set-ups, by schedules complexity preservation (intelligent work organization) and robotised assembly cells (distributed versatility job-shop).

© 2001 by CRC Press LLC

The return on investments deals with leanness, namely on checking each addition or modification on its actual usefulness to increase item's quality; and with economy of scope, namely on carefully monitoring aims and tasks on their ability of granting a positive value-chain, while avoiding unnecessary accomplishments and useless equipments.

These new trends move to intelligent manufacturing set-ups, supporting: recovery flexibility, as option instead of set-apart resources; tactical flexibility, to approach optimal schedules assembly process; and strategic flexibility, for processing variable product mixes. Computer-integrated assembly becomes a cost effective opportunity, whether exploited to draw out actual benefits from the technological versatility of the new resources. Different tracks are considered by the present chapter, namely:

- modularity, to make possible the setting and the resetting of the assembly facility with due account of artefacts evolution, by the off-process management of versatility; and
- robotics, to enable the functional versatility so that several product mixes are processed together, under supervised standard testing operations according to total quality.

The two techniques are each other subsidiary, have different application range, and will be discussed in the following, distinguishing the levels of productivity and of flexibility that each time are needed; sample cases are used for explanatory purposes. Computer-integrated assembly is a relevant aid to fully exploit flexible automation by enabling the process-driven adaptivity, once the visibility on every relevant quantity affecting the processing progression is provided and the transparency of the current decision logic is acknowledged. Such visibility is the preliminary requirement to enable the economy of scope and can actually be reached by expanding the factory automation software with inclusion of proper expert simulation codes.

The present chapter is organised as follows:

- a section recalls the basic options of intelligent manufacturing to support effective assembly processes; namely: the market-driven trends in factory automation, towards computer-integrated assembly, from scientific job allotment to intelligent task assessment; the options of flexibility for achieving cost effective issues; and the basic technologies of the assembly process, from product-oriented assembly lines endowed with special purpose equipment, to operation-oriented functional units based on modular layout or built with robot technology;
- a section considers, at the shop floor level, the characterising features to maximise the return on investments in flexible automation, with focus on assembly problems; i.e., the basic references for evaluating flexibility effects, from the analysis of characteristic features to judgmental hints for the setting of the flexibility figures; the decision tools, supporting the choice of the efficiency setting/fitting figures, exploiting computer-simulation as consultation aid and process-driven govern as controller-manager of shop floor operations;
- a section presents cost effective issues in computer-integrated assembly for situations requiring mass production with sudden switching to new artefacts within short time-to-market terms; namely: the off-process setting of versatility by reconfigurable modular facilities; and the adaptive fitting (recovery flexibility) of buffered modular assembly facilities with (limited) physical resources redundancy; and
- a section presents robot assembly facilities, aimed at exploiting the options of flexibility for customers-driven artefacts; in particular: the on-process setting (strategic flexibility) of robotised assembly facilities; and the efficient fitting (tactical flexibility) by integration of control and management; both situations characterised by the functional redundancy of the knowledge intensive solutions provided by intelligent manufacturing.

The example cases of sections 2.4 and 2.5 have been developed by the Industrial Robot Design Research Group at the University of Genova, Italy, in front of diversified industrial applications at shop floor level, aiming at govern for flexibility issues, according to the basic ideas summarised in sections 2.2 and 2.3.

© 2001 by CRC Press LLC

2.2 Assembly in Intelligent Manufacturing

Efficient manufacturing of industrial artefacts is conditioned by assembly. Product and process reengineering is positively concerned by setting up cost effective facilities. The return on investment is, however, a critical issue; to obtain the right layout, the effectiveness of the assembly section has to be assessed against actual potentialities. The study needs, in general, consider the entire enterprise's organization, from the design to the selling of the artefacts and the degree of automation in both material and data processing has to be acknowledged. At the front-end level one typically deals with:

- fixed assembly stands: the components (suitably assorted and fed) are joined to the (principal) workpieces at properly fixtured stations by, typically, job-shop logistic; and
- transfer assembly lines: the (main) workpieces are transferred by flow shop logistic (with conveyors, belts, etc.) and sequentially joined to the (concurrently fed) parts.

Intermediate solutions, aiming at best compromising effectiveness and adaptivity are, as well, used, e.g.,

- cell shops, performing group technology subassemblies by means of segmented carousels interconnected by adaptive dispatching; and
- transfer sections, joining varying mixes (for adaptive processing, job enrichment, etc.) and enabling several assembly cycles through parts rerouting.

Performance depends on organization and equipment. Productivity (assessed as nominal net production on the reference time horizon) actually reaches the highest figures with flow shop and transfer assembly, based on specially fixtured units (readily adapted to fixed automation) aiming at low cost mass production. Flexibility (related to the property of modifying process abilities to accept varying product mixes) requires technological versatility; job-shop organization with general purpose workstations is prised, in connection to robotics, for on-process and on-line adaptivity.

The emphasis toward product variety, constant quality, higher reliability, frequently updated design, shorter time-to-market, and the likes, forces concurrent enterprise approach, aiming at integrated solutions, from design and development to assembly and delivering stages. Computer integration is the main factor in simultaneously achieving the said goals by knowledge intensive set-ups. Thus, special attention is, for instance, reserved to:

- assembly planning [BAL91], [BOO82], [DeW88], [DEF89], [HEN90], [HoL91], [HoS89], [KoL87], [LeG85], [MAT90], [Mul87], [RoL87], [Wol90];
- design for assembly [AND83], [Bjo89], [BoD84], [BoA92], [Hoe89], [NeW78], [TUR87]; and
- similar options and methods, improving the exploitation of process-embedded knowledge.

Conversely, the layout of the assembly equipment lags behind in flexible automation; as a result, on the final products the related costs percentually appear to increase. For several applications, indeed, robotics in assembly provides meagre benefits, since:

- robots magnify the operation-driven constraints of dedicated equipment (fixtures, jigs, grippers, feeders, etc.) and require a considerable amount of propriety software; as a result, side costs are four to five times the robot cost; and
- manipulation architecture supports poorly optimised motion for any particular task; even if sophisticated path planning and dynamics shaping options are provided, the duty-cycle time and position accuracy are worse than the ones of dedicated units.

An alternative suggests that assembly equipment, built from modular units has to be considered [ACA96a], [Dre82], [GIU91], [MIL93], [Rog93], [TaA81]:

- productivity preserves the figures of special purpose assembly lines;
- reuse of the selected fixed assets into differently configured layouts makes possible amortisation plans based on sequences of product mixes.

© 2001 by CRC Press LLC

The modular approach presumes the interfaces consistency based on mechanical and electronic standards. Then, work cycles are analysed into sets of process primitives having each function performed by a modular unit. The flexibility is managed off-process by reconfiguring the facility as soon as the plans for the mass production of new artefacts are fixed. The opportunity will be considered and example applications are recalled in section 1.4, as issues leading to mass production, while supporting short time-to-market for new artefacts by means of reconfigurability.

Market-Driven Trends in Factory Automation

The availability on the market of comparable offers requires continuous adaptation of current delivery to users' satisfaction, to win new buyers and preserve/expand the trading position of the enterprise. The course turns to become more relevant as the number of specifications is increased to better adapt the products to lifecycle standards on safety, anti-pollution, etc. or on recycling and dismantling rules according to prescriptions aiming at sustainable development promulgated by every industrialised country [AlJ93], [AlL95], [BoA92], [Eba94], [JOV93], [SEL94], [Wie89], [ZUS94]. Effectiveness is dealt by balanced and integrated views: customers' responsiveness, simultaneous product-and-process design, productive decentralisation for technology adaptivity, and the likes. Each offered artefact is, thereafter, endowed by quality ranges attributes covering multitudes of users' requests. Leaving up the mass production aims, the actual trend is to propose (once again after handicrafts time) one-of-a-kind products purposely adapted to individual whims with, however, quality figures granted by standard tolerances, as compared to craftworks, Figure 2.1.

Customised artefact quality is consistent with intelligent manufacturing by means of flexible specialisation. Assembly is a critical step; on-line manual operators are common practice when product variability makes uneasy the facing of changing tasks with high productivity levels. Fully robotised assembly cells

	Craft manufacturing	Mass manufacturing	Customised manufacturing
Enterprise return	economy of skill	economy of scale	economy of scope
Work organisation	master to apprentice indenture	scientific job-allotment	intelligent task assessment
Technical specification	design while manufacturing	off-process optimal assessment	simultaneous product/process design
Decision structure	craftsmen commitment	hierarchical specialisation	decentralised responsibility
Motivation style	individual creativity	division of competencies	collaborative reward
Knowledge features	non replaceable personal contribution	addition of sectorialised team work	distributed cooperative processing

FIGURE 2.1 Trends in market-driven and technology-pushed manufacturing.

© 2001 by CRC Press LLC

are, indeed, endowed by extended versatility so that the mix of items jointly processed can be quite large, but productivity is far from the capability of special purpose assembly facilities. For mass delivery, fixed automation solutions are, therefore, preferred; the switching to new sets of artefacts cannot be done, unless the different special purpose devices, properly matching the requested changes, are enabled. The compression of the time-to-market is sought by simultaneous engineering, namely, by developing products (with design-for-assembly, etc. rules) and processes (with modular configurability or function programmability).

The ameliorations have been related to the automatic preparation of the assembly sequences [ArG88], [Bon90], [Boo87], [JIA88], [Koj90], [MIC88], [Tip69], [WAR87] with attention focused on the process modelling, aiming at plant fitting for granting the visibility of every relevant effect. Formal descriptions have been likely proposed to design the assembly facilities [ACA96B], [ArG88], [Hoe89], [LeG85], [Moi88], [ONO93], [ONO94b], [SEK83], [Van90], [WIE91] liable to be translated into functional models. The computer integration is a powerful contrivance; centrality, however, shall be left to the manufacturing flow, according to requirements of leanness, which say that non-necessary options are directly (since not fully exploited) and indirectly (because redundant accomplishment) nuisances.

Summing up, important improvements are expected from the integrated control and management of the processing operations [ACA88c], [ACA89c], [ACA92b], [MIC92b], [MIC94d], with account of flexibility of the physical (the set-up) and the logical (the fit-out) resources. One should look for:

- the effective set-up of assembly sections, tailored to product mixes included by the enterprise strategic planning;
- the proper fit-out of assembly schedules, adapted to production agendas within the enterprise tactical planning.

Off-process and on-process adaptivity, Figure 2.2, happens to become a market-driven request; it has to be tackled over at the proper level:

- setting is concerned by the structural frames, Components, facility-configuration and control: CFC the set-up of CFC frames presents as everlasting activity; choices provide reference for identifying current process set-ups all along the life of the facility; and

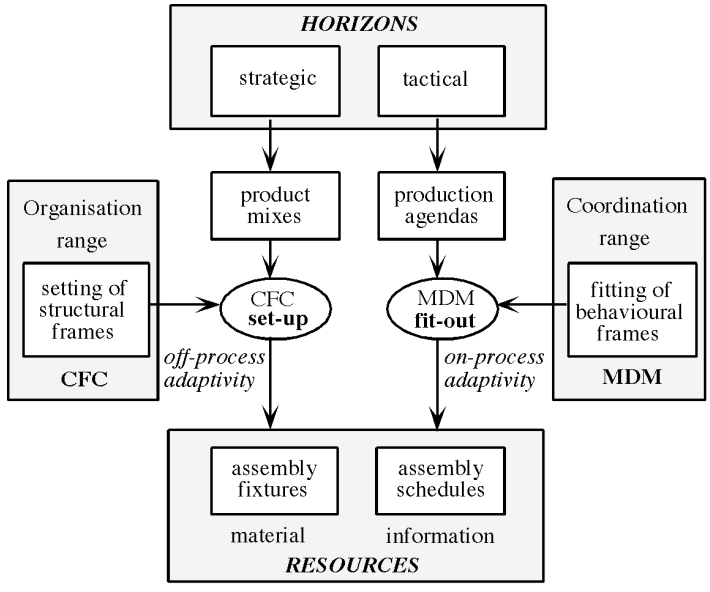


FIGURE 2.2 Flexibility setting/fitting by controllers/managers.

© 2001 by CRC Press LLC

- fitting deals with information options of the behavioural frames, Monitoring, decision-manifold and management: MDM the MDM frames, by acknowledging the plant operational states and functional trends, offer data for the on-process improvement of the efficiency.

Technical Analysis of the Return on Investments

Leanness is suitably related to the monitoring of the value added to products by each investment into new physical or logical resources; computer-integrated assembly looks for cost effective set-ups aiming at economy of scope by means of a knowledge intensive frame, purposely restricted to a series of rules, such as:

- to extend product mix variability to agree with larger amounts of consumers' wishes;
- to avoid investment in special rigs and exploit robotisation for diversified products;
- to limit inventory and enable adaptive, bottom-up, just-in-time schedules;
- to suppress redundancies and set-apart resources and instead apply recovery flexibility;
- to abolish not strictly necessary functions and use decentralised responsibility;
- to exclude sectorialisation of competencies to solve problems where they arise;
- to enhance customers' driven responsiveness by minimal time-to-market; and
- to exploit involvement, for improving products and processes by shared interest.

The different situations asserting economical returns encompass:

- the exploitation of wide-versatility facilities, assuring simultaneous manufacturing of extended mixes of products, conveniently distributed within the work-shifts;
- the use of system integrators granting one-of-a-kind customised products delivery, by incorporation of parts or subgroups provided by specialised suppliers;
- the resort to modular assembly facilities, with by-passes and buffers between special purpose units to perform the adaptive scheduling of moderately varying delivery; and
- the establishment of multiple reconfigurability lines, joined to the concurrent design of products and processes, to reach step-wise tracking of customers' satisfaction.

The presetting of the economically effective solutions has been investigated from several stand-points [ACA89F], [ACA95], [Beh86], [Eve93], [Gus84], [Tan83], [TOB93], [WaW82]. The rentability of the (off-process) setting and of the (on-process) fitting for govern for flexibility issues, is, mainly, assessed by computer simulation. The choice of the appropriate set-ups refers to a few factors:

- investment costs, with amortisation plans within the production programmes;
- productivity performances, to grant technical specifications and trade goals;
- delivery figures (or time-to-market), according to customers' expectation; and
- quality (fitness for purposes and conformance to specifications).

Time-to-market and artefact quality are important factors for enterprises aiming at remaining, or becoming, worldwide competitors. With markets globalisation, a factory cannot be sure to propagate its protected trade segmentation; short delivering with customer-driven quality is therefore becoming a critical request [Beh86], [Cuc89], [Lil66], [MIC89], [MIC92c], [MIC94d], [MIC95a], [MuY90], [Nar81], [StB93], [Tak91], [TOB93], [Wie89].

The actual effectiveness is still an open problem and convenient opportunities to go a little further seem to be related to the ability of exploiting, to the best, the empirical knowledge acquired on the field. Craftiness and training are widely trusted in manual assembly. The translation into automated assembly tasks is obtained by specialisation (frosting off-process the knowledge). Knowledge-based approach and expert simulation are opportunities to assess flexibility by uniformly combining causal and judgmental knowledge; they are offered to production engineers for making possible:

© 2001 by CRC Press LLC

	Structures	Ranges	Horizons
	Framings	***Actions***	***Functions***
Supervisory level	Enterprise	Organisation	Strategic
	Setting	*Planning*	*Management*
Communication level	Facility	Coordination	Tactical
	Fitting	*Scheduling*	*Control*
Operational level	Device	Execution	Recovery
	Monitoring	*Enabling*	*Command*

FIGURE 2.3 Hierarchical information layout.

- the optimal setting of products and processes, according to the rules of simultaneous engineering; and
- the best return on investments, by preserving leanness into intelligent (knowledge intensive) manufacturing.

Choice of Resources and Technical Options

Once market goals are acknowledged, the computer-integration looks for the most effective setting of the resources avoiding dis-economies due to exaggerated sternness in work organisation. To such a purpose, flexibility in manufacturing is distinguished by range and horizon, usually enabling hierarchical information layouts, Figure 2.3, so that:

- at the organization range, the overall production agendas are planned, according to the enterprise policy, over the established strategic horizon;
- at the coordination range, the selected products mix is scheduled, for maximising pieces delivery on the proper tactical horizon;
- at the execution range, the discontinuities (at unexpected or planned occurrences) are overridden within the recovery horizon.

The assembly lines, originally conceived for mass production, can be endowed with flexibility by synchronising assortment of parts and processed workpiece. The set-ups extensively exploit human workers directly on-process, with schedules cadenced by the feeding services. The solution is consistent with (Taylor) scientific work organisation [MIC94a] based on the job allotment paradigm and on the three-S constraint simplify, specialise, standardise. By that way, the assembly cycles are properly optimised off-process with specification of each elemental operation so that no ambiguity could be left to front-end operators. The final product is granted to be released within tolerated quality, provided that nothing is moved off the preset plans.

The fixed automation issue requires little changes in the artefacts design (e.g., reduction of components number, enhancement of unidirectional parts feeding, etc.), and the assessment to figure out the investment amortisation plans is straightforward, once productivity and overall delivery are achieved. Aiming at economy of scale, the assembly automation was confined at developing dedicated special purpose fixtures; by switching to economy of scope [MIC92a], [MIC97], [MiR96], the inclusion of suitable flexibility is needed, to exploit adaptive scheduling and recovery abilities. Variability of workpieces requires similar variability of parts to be joined; an acceptable arrangement makes use of preassorted kits of parts, forwarded along programmable transfer paths, crossing the workpieces flow. The arrangement

© 2001 by CRC Press LLC

is exploited, for instance, by car manufacturers when several models are assembled on the same line. The planning of the local joining stations presumes monitoring and diagnosis operations to be performed on-process; the actually delivered products need be scheduled aiming at just-in-time policies related to customers' requests. The planning aspects have already been extensively investigated with focus on the integration level of the manufacturing activities [BAL91], [DEF89], [DiS92], [Din93], [HEN90], [KAN93], [LaE92], [Mar89], [SaD92], [SAN95], [SEK83], [Van90], [WAA92], [WAR87].

The different set-ups, performing the automatic assembly of wide mixes of products, need combine the versatility of the joining units with the adaptivity of the material logistics; investment in fixtures needs be motivated by higher productivity and quality. The choice of the assembly facility requires the previous assessment of its efficiency; the result is achieved by functional models and computer simulation: these provide accurate descriptions of the relevant transformation affecting the material processes and of the governing logic that might be enabled for exploiting the resources on the (different) execution, coordination, and organisational horizons. The assembly phase deserves increasing relevance and computer integration develops as critical issue with several possible hints and pieces of advice for the on-process and on-line exploitation of flexibility ranging at the different functional ranges and operation horizons.

The work organisation is, thereafter, concerned by changes in progression, aiming at back inclusion of decision manifolds [ACA87a], [ACA88b], [ACA89b], [ACA89e], [ACA93], [MIC90], [MIC94b], in order that by adaptivity, the optimal running set-ups and fit-outs are continuously redintegrated and made ready to exploit the available resources according to the enterprise's policy. Intelligent manufacturing, therefore, is based on incorporating robot technology as front-end equipment and expert governors for tasks scheduling of time-varying production plans [ACA86], [ACA87b], [ACA87d], [ACA89b], [ACA89c], [ACA89e], [ACA92b], [MIC89], [MIC90], [MIC92b], [MIC94b]. Programming, in front of high variability mixes, looks for job-shop organisations with robotic assembly stands or cells [ArG85], [AZU88], [Bon90], [Kir92], [Lot86], [MaF82], [MOL92], [NoR90], [SCH90], [StB93], [Tak91], [TAM93], [UNO84], [VaV89], so that the technological versatility of the installed equipment can face changing situations provided that shop logistics grant the correct transportation and feeding of parts and fixtures.

Aiming at intelligent manufacturing, the three-S constraints approach is replaced by the three-R option, namely: **r**obotise, **r**egulate, **r**edintegrate, so that:

- robotisation is based on flexible automation enabled by the technological versatility of the equipment to support fitness for purpose innovation;
- regulation is concerned by condition monitoring maintenance to uniformly obtain conformance to specifications within the principal process; and
- redintegration presumes the redesign of, both products and processes, by preserving complexity for market-driven quality enhancement.

The three-R paradigm modifies the scientific job allotment precepts into the new intelligent task-assessment rules; thereafter:

- redintegration grants that, by quality engineering, the visibility of conditioning facts makes possible to keep what delivered to perfect (according to specifications) state;
- regulation means that commands are operating on-line for adapting/restoring the process depending on the (changing) products mixes; and
- robotisation is here understood as the ability of giving decision supports based on the on-process knowledge instensive environments.

Flexible automation develops with technological innovation in material processing fixtures, for dispatching and transform operations and, in parallel, in the information processing aids, for monitoring and govern operations. The three-R paradigm leads to new trends in work organisation aiming at intelligent task assessment; intelligence presumes that complexity shall be faced dynamically, since

© 2001 by CRC Press LLC

analysis generates time varying elements and cannot lead to frozen plans; and optimal schedules evolve along with the process and only (higher level) tasks are useful addresses for preserving the enterprises' effectiveness.

Cost Effectiveness by Means of Flexibility

Improvement of performance depends on exploiting plant flexibility. The goal takes principal part in widening product mix variability and critical role in avoiding idle resources. Return on investment arises from sets of rules, expressing the objectives of complexity preservation by means of intelligent task assessment, namely:

- functional integration along the principal manufacturing process, to support the synergetic cooperation of every factory resource;
- total quality, for globally conditioning the enterprise organisation to be customers' driven, by incorporating the fitness for purpose as artefact feature;
- flexible specialisation, to assure intensive exploitation of facilities by expanding the offered mix, through technological integration and productive decentralisation;
- lean engineering, to avoid redundancy and minimise investment and personnel, in relation to the planned production requirements over the enterprise strategic horizon.

Assembly is a challenging goal due to task complexity; the issues cannot be disjoined from expected returns. Effectiveness is a combined outcome of specialisation (three-S aims) and flexibility (three-R options) and needs be assessed by standardised references [ACA95], [ACA96a], [BAL91], [Beh86], [Eve93], [Gus84], [KOJ94], [Mak93], [MIC94d], [MIY86], [MIY88a], [MIY88b], [ONO94b], [SHI95], [TOB93], [WIE91]. Flexibility effects are particularly relevant at shop floor level and the discussion will focus on such kind of problems. Achieving flexibility depends, of course, on the initially preset layouts and facilities; granting return on investments is, moreover, widely dependent on the govern for flexibility adaptive exploitation of plant and process. Improvements are obtained by iterating a decision loop, which refers to a functional model of the facility behaviour and is validated by supports based on the measurement of plant performance and the comparison of current figures against the expected levels of efficiency. Such decision logic is effective on the condition that every alternative matching the application case is investigated; this is feasible through simulation, granting virtual reality display of actual production plans. Indeed, it is not possible to preset and implement versatility as enterprise policy and to control and manage adaptivity as current request unless the effects are measured and the facilities are tuned to flexibility.

Expert simulation is profitably used at the design stage for a beforehand evaluation of alternative facilities; it becomes permanent consultation aid, during exploitation, to select or to restore the best choices. The goals are achieved by means of specialised software for factory automation, that incorporates AI aids; example implementations are recalled in Figure 2.4 [MIC94c], [MIC95b], [MIC96a], [MIC97]. The decision cycle for setting/fitting flexibility aims at economy of scope, by changes in work organisation first acknowledged by Japanese enterprises. In fact, when Japanese and Western countries enterprises are compared, differences in effectiveness are found in the interlacing of material and information flows, that entails knowledge distribution and decision decentralisation issues and mainly leads to:

- piece wise continuous betterment: to yield the successful effort of adapting products to consumers' wishes (increasing quality and lowering price);
- diagnostics and monitoring maintenance: to aim at company-wide quality control and at predictive maintenance policies;
- cooperative knowledge processing: to enable a reward system granting individual and team creativity, which aims at innovating products and processes; and
- lean engineering check-up assessment: to exploit value-chain models and remove material or information additions, that do not improve enterprise profitability.

© 2001 by CRC Press LLC

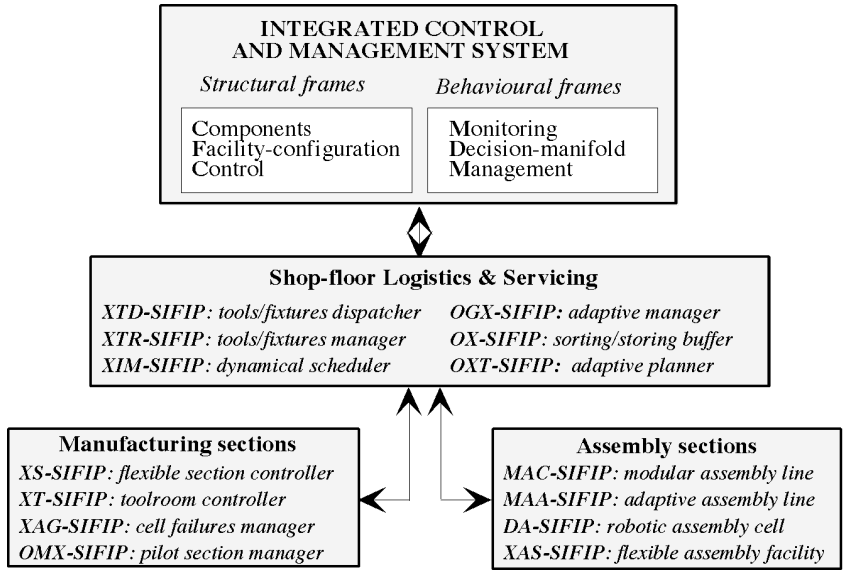


FIGURE 2.4 Example software packages for the integrated control/management of flexibility.

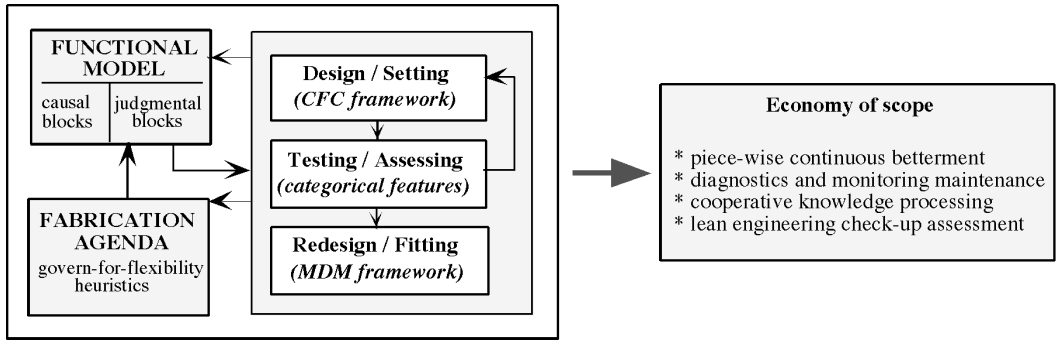


FIGURE 2.5 Decision cycle for setting/fitting flexibility

The four issues are equivalent views of dynamical allocation of tasks, by intelligent preservation of complexity, into facility set-ups (granting return on investment by adaptive scheduled delivery) and into process fit-outs (aiming at highest effectiveness, by recovery flexibility). Goals are addressed by recurrent procedures, using distributed knowledge processing schemes, to move back on-process the decisional manifold consistent with flexible manufacturing. Two procedures distinguish:

- one for acknowledging the suitability of the preset assembling layout; and
- one for supporting the operativity of the govern-for-flexibility options.

Design and Exploitation of Flexible Assembly Facilities

The development of assembly fixtures, incorporating proper functional flexibility and aiming at cost effectiveness by means of economy of scope, is based, Figure 2.5, on the iteration of the three steps: design/setting; testing/assessing; redesign/fitting. The cycle illustrates the interactive nature of decision making and shows how the behaviour of an alternative influences which alternatives are identified for the next loop.

The application of decision cycle models to intelligent assembly is concerned with the issues of governing flexibility, so that varying market-driven requests are satisfied whenever they emerge by bottom-up plans. Demanding aspect is that flexible plants are not used as they are, rather after setting

© 2001 by CRC Press LLC

and fitting, according to data collected by testing the issues of market-driven alternatives. The presetting of the versatility requirements during the development stage of the assembly facility and the on-process exploitation of the functional flexibility are better explained with the help of example developments. This is done in sections 2.4 and 2.5 of this chapter.

For the set-up of assembly facilities properly assuring the return on investments, the feedback in capacity allocation decision cycle is built by monitoring the flexibility effects; it provides reference data for the overall procedure. The step measures the effectiveness of competing options by means of the assembly functional model with the use of virtual reality; real assembly facilities cannot be used since, at this stage of the development, they are not yet properly set; the use of pilot plants is restrictive with, most of the times, unrealistic constraints. The locally most efficient alternative is recognised at the last step in the decision loop. Each choice represents an optimised configuration of the flexible capacity allocation and an effective governing policy, with constraints on the shop floor operations, such as adjust the routing table, adjust the products schedule, modify the assembly duty plan, modify the inventory policy, and the likes.

At the first step, the preliminary setting of the assembly facility is prepared to match the case requirements and is fixed referring to past experiences. A detailed functional model of the facility is dressed, implementing causal blocks for simulating the transformations that physical resources undergo, and judgmental blocks for emulating the govern schemes that logical resources enable.

At the second step, the testing is mainly performed by simulation. Causal inference provides the assessment of performance by means of categorical features (patterns of actions); heuristics is called for to implement govern-for-flexibility procedures. The net production figures are evaluated with the considered fabrication agendas for several schedules and operation conditions. Cross-coupling effects are common occurrence for time-varying, low inventory, lean manufacturing set-ups. The effects of flexibility have an impact on the net production in such a way that factory output depends on several factors and not only on the marginal efficiency of the engaged resources.

Finally, the understanding of flexibility effects is used for choosing, at the third step, the CFC and MDM frames. Through iteration, statistical inference can be used to build up structured databases, back exploited to orient choices and to redesign the plant by patterns of structure. Reference data continuously evolve, providing structures-and-actions patterns so that enterprises could be set/fit into position of tracking economy of scope conditions. For assembly upgraded fitting, the decision loop starts by the issues of managing flexibility, over the selected horizon (strategic, tactical, and operational), with the suitable enabling logic: dispatching decision, schedule decision, planning decision, inventory decision, etc. Structures and horizons cannot be acknowledged separately. On these premises, suitable fabrication agendas are planned, specifying:

- the product batches (part assortments, work cycles, shop logistics, delivery dates, etc.) and the related tactical schedules;
- the batch size and sequencing policy, according to customers' requests, with criteria for managing transients, also at one-of-a-kind production;
- the maintenance and restoring plans, with indication of the monitored signatures and the risk thresholds; and
- the likes.

Results, properly established by simulation, provide a uniform basis to compare (for each CFC frame) the economic benefits enabled (and acknowledged) by means of the pertinent MDM frame. Computer aids should include the ability of recursively running the setting, testing and fitting steps according to the outlined decision loop.

The Technology of the Assembly Process

The manufacturing segment concerned by assembly is the process by which parts are mated into a product. This could be a subgroup, to be shipped for further assembly, and a similar analysis starts again. The basic analysis, therefore, consider a workpiece and several parts to be joined. Typically, for assembly

© 2001 by CRC Press LLC

lines, workpieces are palletised; possibly, a single pallet carries more than one piece to be joined with the related parts; most of the times, pieces are addressed by considering the carrying pallet. The assembly posture can be moving with continuous transfer, coordinated with the parts feeding cycles, asynchronously related to the assembly cycles, or stops at a given station (for limited batch production). When number of items is large, amount of workpieces is small, requested productivity is high and complexity of parts is critical, transfer lines could profitably be replaced by a sequence of carousel assembly stations suitably interconnected by transportation systems.

Assembly requires dextrous training and sophisticated skill. Efficiency was sought, in the past, with the three-S constraints (**s**implify, **s**pecialise, and **s**tandardise), leading to scientific job allotment. It is presently approached through the three-R options (**r**obotisation, **r**egulation, and **r**edintegration), presuming intelligent task assessment. This is consistent with the knowledge intensive organisation of intelligent manufacturing; it requires functional characterisation of the physical transformations and behavioural description of the decision frames exploited for planning. Computer integration is an achievement according to twofold trends:

- as arrival issue of flexible assembly lines in factory automation, with knowledge processing management supporting total quality as standard opportunity; and
- as advanced issue of adaptive work-stands, based on the robotic assembly of extended mix of products through integrated control and management.

Solutions differ for cost effectiveness depending on the case applications and facilities should be correspondingly adapted. Short comments on the reference technologies are recalled, always remaining in the field of automatic assembly.

The automation of an assembly process requires the previous specification of every elemental operation; the overall task splits on sets of facilities usually arranged as:

- assembly line, namely, a sequence of postures with concurrent parts feeding systems and dedicated joining devices; and
- assembly stand, namely, a workstation with piece placing and parts feeding fixtures, pick-and-place mechanisms, and joining units.

For flexibility, the facilities turn to be classified according to the built-up options:

- assembly section: (typically) a sequence of carousel assembly stations with related parts and fixtures logistics and robotics; and
- assembly cell: a servoed and fixtured posture, with appropriate feeding systems and one (or more) instrumental robots.

The two sets of assembly organisations are reviewed, with introductory hints for the case discussions of the following sections.

Product-Oriented Assembly Lines

When the production volumes are high, the assembly line concept remains the most effective reference, Figure 2.6. The basic flow of products is associated to the transfer of workpieces (or pallets), from an initial station where the reference pieces are loaded, to the final station where the products are delivered (for subsequent processing, for packing operations, etc.). The assembling is, typically, operated with workpieces flow carried by:

- a conveyor chain or continuous transfer: the workpieces, with the proper parallel parts feeding, are tracked by the joining devices; and
- an intermittent chain or indexing transfer: the workpieces stop to have the parts feeding and the joining operations are performed at fixed time intervals.

© 2001 by CRC Press LLC

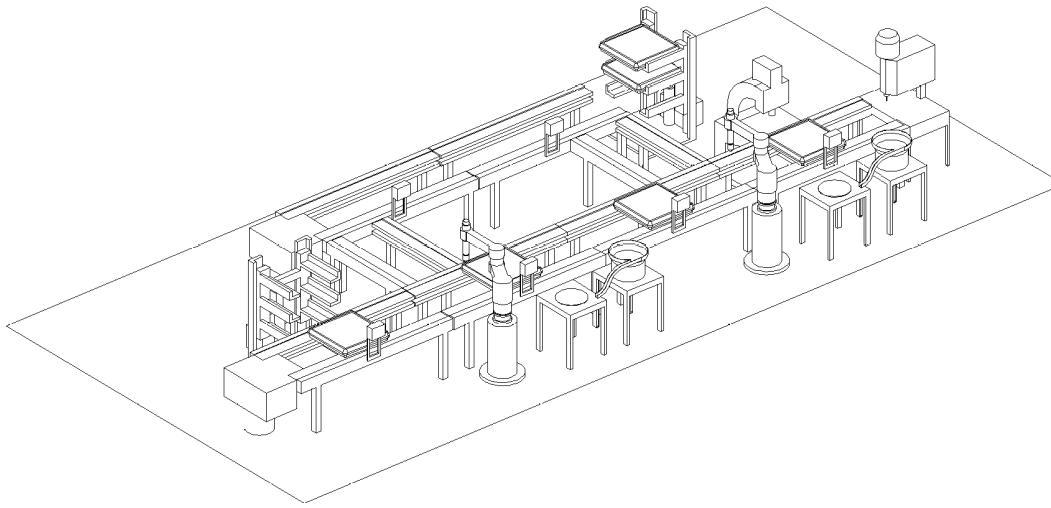


FIGURE 2.6 Layout of a typical assembly line.

For the present analysis, palletised pieces or simple workpieces directly carried by the chain links or belt elements are indifferently addressed as unit items. It should only be reminded that, with palletised transportation, the loading station is connected to the corresponding pieces fixturing area and the unloading station is followed by the products delivery area. Size and shape of pallets depend on the products; the number of pallets is minimal for conveyor belt transfers. For practical purposes, it is assumed greater than twice the number of postures, for the indexing transfers; and it approaches an equivalent amount, referring to the moving postures of conveyors.

Figure 2.6 shows how the assembly line can be obtained by combining suitable functional units (and related part-feeders) with pieces positioning and handling rigs. These bear an array of pallets, with synchronised, cyclic travel. Workpieces are positioned and fastened by a picking and latching unit; an unlatching unit delivers the pieces to the belt for transportation to the next processing posture. Part assembly is done by a single stroke; the two steps cycle "positioning + joining" requires two work strokes. Each station is fed (with FIFO policy) by the conveyor with interoperational buffering capability; the duty cycle repeats the scheduled actions, namely:

- withdrawal of half-finished workpieces from the input station and palletisation;
- check of components presence, location, and positioning, to initialising the job;
- execution of the scheduled assembly/testing actions (by single/multiple work strokes);
- acknowledgement of tests (removal of defective workpieces, whether recognised); and
- de-palletisation and delivery of the (tested) workpieces to the output station.

Sequencing and coordination is granted by means of the pallets carousel, discontinued by stops for doing the listed actions; the longest interval defines the halt span, which is as short as the unit/multiple stroke of fixed automation. The number of pieces fastened to pallets can be modified so that all stations approach balanced cycle conditions.

Continuous transfers have been using front-end operators; they are now exploiting tracking robots. Indexing transfers can split into a series of (automatic) assembly cells with local revolving tables and clusters of joining units. The productivity is related to the ability of minimising the individual duty cycle times and of paralleling the elemental tasks, so that net production (per shift) is inversely proportional to the (average) unit duty cycle. Optimality means the careful setting of the assembly schedules and the proper development of special purpose joining devices. Once the assembly fixture is enabled, the least disturbance on the process continuity is a penalty to the pre-programmed (optimal) schedule.

© 2001 by CRC Press LLC

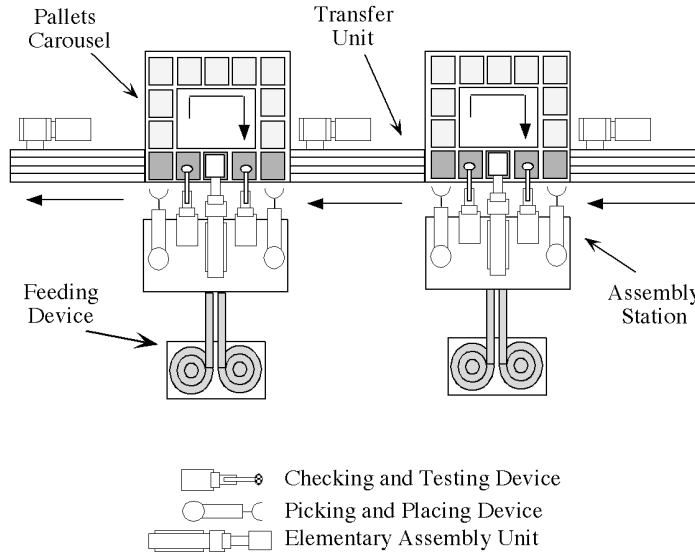


FIGURE 2.7 Modular assembly line with local palletisation.

The soundness of the line is critical; the assembling stops, whether a single intermediate units becomes defective.

Computer integration is the basic option to keep visibility on the process variables. Looking for flexibility, however, the preservation of the conventional architecture is doubtful, indeed:

- versatile robots, replacing special purpose assembly units, require trimming fit-up and lower efficiency is commonly reached with higher investments; and
- shop-logistic faces severe planning troubles for part feeding, with diffused complex storing and sorting services, having questionable reliability.

Then, long assembly lines are better split into short loops with interconnecting tracks to be timely enabled according to the schedules. The layouts are convenient for a comparatively large variety of items to be assembled by the same plant, into batches, with single items modified in function of the customers' orders. The facility requires comparatively high investments and returns are obtained when the overall products mix reaches the expected volume on the planned horizon.

The flexibility, however, is being considered with increasing interest by worldwide enterprises, to respond by customers' driven new products, with the shortest time-to-market. Resort to modular assembly facilities is, perhaps, a winning opportunity, Figure 2.7. The layout can be conceived once a preliminary investigation is carried over for analysing the reference operations and for acknowledging the related sets of:

- shop-logistic modules, for buffering, transfer, and handling along the principal flow;
- parts feeding modules, for granting storing and sorting services of secondary flows;
- processing modules, such as: screw-drivers, joining-fixtures, clinching-rigs, etc.;
- fitting modules, to assure compatibility among the resources interconnection;
- testing modules, for supervision, quality control, and coordination duties; and
- govern modules, for supporting and controlling task-adaptive programming.

Any given assembly set-up behaves as a special purpose facility properly adapted to the considered product types. When the market asks for new items, the enterprise starts to jointly design product-and-process; the reconfiguration of the facility makes possible to assemble the new items by differently combining the same functional modules. Investments are amortised over several product types in series, and, to some extent, with no time bounds. Modularity is consistent with several manufacturing segments, with functional units

© 2001 by CRC Press LLC

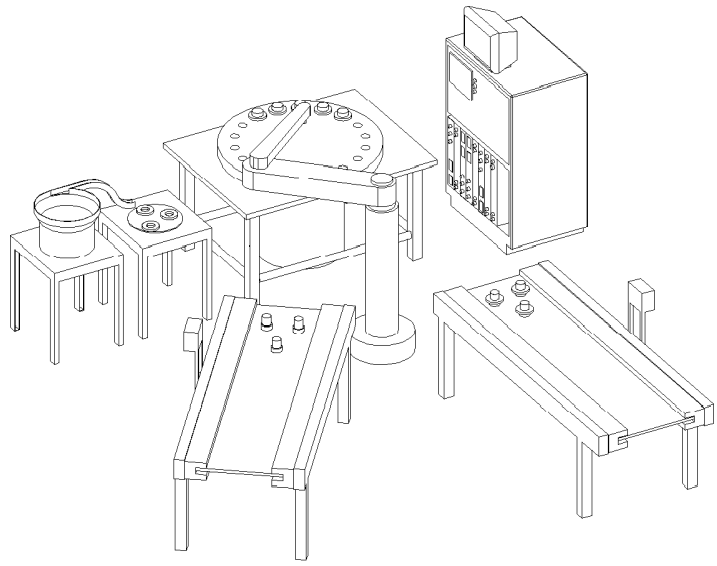


FIGURE 2.8 Layout of a typical robot-operated assembly cell.

having high productivity standards; a typical development is discussed as first example application of section 2.4. Modularity, furthermore, provides hints to add on-process flexibility by managing redundancy, instead of versatility; such option is considered as second example application of the same section 2.4.

Robotic Assembly Cells

When products variability is very high and the joining units cannot be identified by constant technical specifications, robotic assembly is the technology-driven option to replace the on-line operators, Figure 2.8. By referring to the workpieces supply, one distinguishes the assembling performed at:

- fixed stands, connected by a programmed transport service (AGV operated, etc.): the parts to be joined are commonly fetched with random access order; and
- free cycles postures, supplied by a process-supervised discontinuous transfer service: the parts are, possibly, assorted into kits or fed with ordered sequences.

Continuous motion or indexing transfer cannot generally provide proper adaptivity to deal with secondary flows and duty-cycle variability along the principal material flow. The secondary material flows are represented by the feeding of items to be joined to the workpiece. This parts feeding requires the (elemental) functions of procurement, store up, singling, orientation, dispatch, and escapment. Small size parts are processed by vibratory bowl feeders with horizontal or vertical forwarding. Assorted parts are conveniently fed on trays; belts or AGV transportation systems are used for the tray dispatching and an important aspect covers parts orientation and setting on the trays. A more elaborated solution uses parts kits and suitable kit fixtures separately fixed at suitable storing and sorting stations.

The fixed stands can be endowed with a revolving table. The free cycles postures are often arranged over a loop; workpieces might, eventually, be brought more than once at the same location. The coordination of parts feeding can be performed at the stands by frontend schedulers or can be managed at shopfloor level by a supervisor-driven sorting and dispatching service between the postures. Programmable robotic units can execute several duty cycles, provided that the proper fixturing options are supplied.

The assembly at fixed stands is common practice when varying parts are joined to difficult to handle workpieces, because rather complex or excessively heavy. Large series of items, to be simultaneously delivered, requires multiplication of stands. The solution is, on the contrary, consistent with one-of-a-kind production and is a worthwhile assembly opportunity for a flexible specialisation supplier offering

© 2001 by CRC Press LLC

co-design help to system integration businesses aiming at customised final products, as it is discussed by the first example application of section 2.5.

The assembly at free cycles postures can be seen as a further splitting of the timely enabled transfer loops up to job-shop planning, with suitable parts dispatching service. The solution is consistent with manufacturing a highly variable mix of products, that need be delivered within short due dates with changing properties and assortments. Several scheduling updatings shall be performed on-process to restore high efficiency production plans (granted by tactical flexibility) in front of programmed (due to the strategic flexibility) or unexpected (faced by execution flexibility) occurrences. Details on this kind of development are summarised in section 2.5, as second case example.

2.3 Effectiveness Through Flexibility Issues

To understand factory performance in connection to flexibility, the relations between production variables need be stated in actual running conditions and assessed by figures giving the net production as this results for real plants [ACA86], [ACA93], [ACA95], [ArG88], [Beh86], [Mak74], [MAK93], [MIC94a], [MIC94c], [MIC94d], [MIC95a], [SCH90], [Son90], [VaV89], [VOH92]. The net production, rather than stand-alone quantity, should be related to experimental properties such as manufacturing efficiency, production capacity, and cross-coupling effects. Basic concepts are recalled.

The manufacturing efficiency is obtained from variables that have an inverse impact on productivity, namely availability of machines, labour, scrap percentage, and set-up delay. Their dependent nature makes partial figures useless; for efficiency, the coupled effects need be mastered. Efficiency is improved by appropriate techniques, such as trend monitoring pro-active maintenance, anthropocentric work organization, total quality, on-line process control, automation, self-adapting fixtures, and the like.

The production capacity is the measure of the gross number of artefacts which can be produced within the planning horizon. It is not a static figure: any products mix (instance of production requirement) establishes a production capacity; whenever the mix changes, so will the production capacity. Its value is computed, for given planning horizon, in function of production requirements (fabrication agendas, etc.), process plans (list of acquired jobs, work time standards, etc.), amount of resources (number of machines, tools and operators, etc.), and other updated information on the process.

The cross-coupling effects are losses in production capacity, in running conditions, due to combined occurrences that make beforehand established schedules unfeasible, such as resources shortage and process blocking. The issue is explained by the example occurrence: material shortage and bottleneck propagation; stops of parts feeding have downstream outcomes with delay on assembly and delivering, and upstream upshots with obstruction formation and stacking of parts between workstations.

Performance figures and coupled effects of flexibility are empirical data obtained by experimenting on real plants or, through functional models, by computer simulation. Effectiveness, in the practice, is enabled by economy of scope as previously said, with leanness check-up assessments; the view is, perhaps, reductive since progresses are consistently stated by the knowledge intensive concurrent enterprise [MIR96], through the integration, Figure 2.9, of marketing, design, manufacturing, and finance systems. To reach more comprehensive assessments, issues are stated by collecting and comparing properly standardised indices and by tracking and acknowledging properly established methods. Concurrent enterprise concept is valuable frame to help combining the marketing, design, manufacturing, and finance activities into unified data structures, travelled by knowledge along clustered computers, as servoed support of the production processes.

In this section, proper indices and methods are considered turning company-wide aims into the specialised segment of artefact's assembly. Even within the said limits, the knowledge architecture is not trivial since tasks (product design, process planning, scheduling, shop-control, material handling, joining sequence, quality check, artefact delivering, marketing, sales management, etc.) need simultaneously be considered, with due account of requirements for the location scattering of processed materials and time synchronisation of planned operations. The architecture should enable data-file exchange (to distribute

© 2001 by CRC Press LLC

CORPORATE FUNCTIONS	CORPORATE ACTIVITIES
MARKETING SYSTEMS	• order processing: track delivery • record of sales data: sales region analysis • market analysis: customers identification • pricing analysis: sales trends of products • trade analysis: market share estimate
DESIGN SYSTEMS	• conceptualising: selection of new products • product designing: technical specification files • engineering: throughput/resources programming • quality assessment: customers' satisfaction • re-engineering: value-chain modelling
MANUFACTURING SYSTEMS	• plant coordination: investments, labour, throughput • production programming: jobs allocation, master plans • resources allotment: facilities setting and fitting • process planning: material procuration, inventory control
FINANCE SYSTEMS	• capital investment analysis: long-terms forecast • fixed assets planning: resources allocation policy • cash management: track receipts, determine purchasing • budgeting: prepare short-terms schedules • accounting and payroll: maintain records

FIGURE 2.9 Concurrent-enterprise knowledge-intensive set-up.

information to decentralised processing units) and decision aid activation (to share intelligence between the local knowledge-based systems).

The resulting layout supports the controlled collaboration of intelligent manufacturing and, mainly, two philosophies have been considered:

- hierarchical structures, based on distributed problem solving abilities; and
- multi-agent systems, with message passing management.

The communication model is centralised in the first case; distributed, in the second. Intermediate combinations have also been developed. On-line flexibility is, however, enabled either ways; the multi-agent architecture enhances reset and outfit flexibility since independent entities are defined to acknowledge the different jobs.

Deep investigation on single stages (here, the assembly process) avails itself of:

- empirical scales, assessed, according to the representational theory of measurement, as mapping standard of the plant performance into characterising indices; and
- functional models, established, after task decomposition, by duty sharing between agents, as reference method for computer simulation/emulation.

The assembling activities are, thereafter, faced by integrated design and manufacturing with developments moved on from the water-fall style, adding iterative loops within and among the different phases and performing the interlaced specification of products and processes, Figure 2.10.

The next two paragraphs of the section are, respectively, devoted to the description of the influence quantities (and of the related empirical scales) that modify the setting of the functional ranges of plant flexibility, and to the presentation of the computer aids required for the evaluation of the flexibility

© 2001 by CRC Press LLC

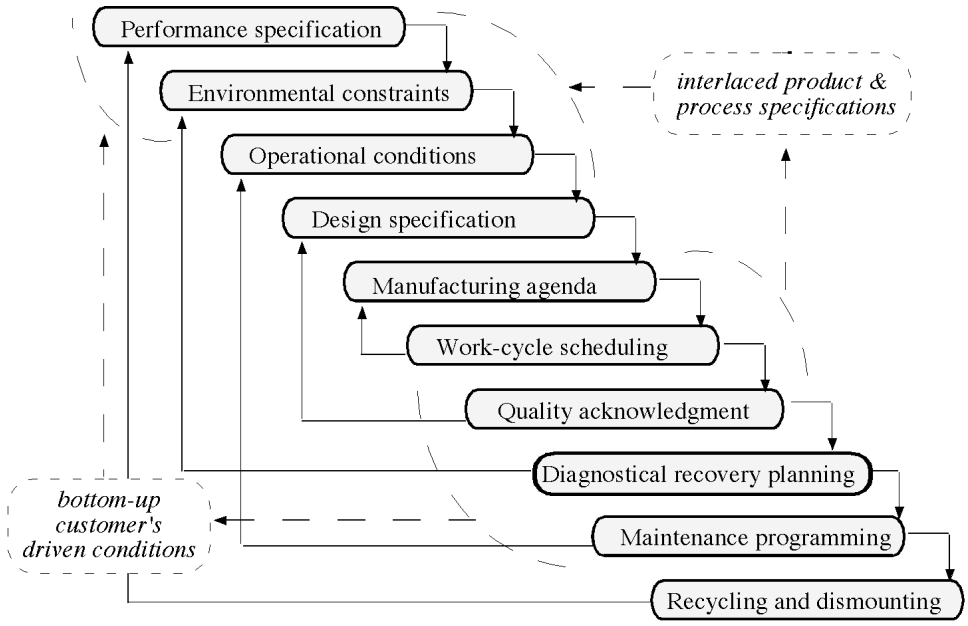


FIGURE 2.10 Water-fall style presentation of the integrated design activities.

effects by emulation/simulation of the plants actual behaviour (through consistent functional models). The last paragraph introduces to the example studies, discussed in the next two sections of the chapter, which typify flexibility situations by means of existing industrial facilities with the support of specialised software purposely developed for each case.

Assessment of the Flexibility Requirements

The functional ranges of flexibility preserve common characterisation all along the intelligent manufacturing process. The backup knowledge architecture presents with hierarchic set-ups, reflecting differences on physical facilities and, as deeply, on logical resources; this leads to specialised aids, such as:

- expert consultants, aiming at the strategic management of the enterprise policy;
- distributed supervisors, for continuous monitoring and coordination of facilities; and
- localised actors, performing peripheral diagnostic and task sequencing execution.

The recalled solutions distinguish the range of flexibility, Figure 2.11, namely:

- the upper range aims at the effectiveness of the economy of scope, by enabling, after simulation-based investigation, supervising managers, with a balanced utilisation of the strategic flexibility along with the organisational range;
- the intermediate can approach the productivity of special purpose automation, once proper set-ups are configured, to make feasible optimal schedules ruled by on-line distributed controllers along the coordination level at the tactical flexibility range;
- the lower exploits robot versatility to grant manufacturing continuity by executional flexibility ruled by adaptive commands; productivity lowers, but production continuity is restored and products delivery is preserved at planned or at unexpected occurrences.

The general precepts leading to effectiveness by flexibility have been summarised earlier in this Section; the assessment of assembling characteristics is hereafter shown defining a set of elemental quantities that help dressing a comparative evaluation of competing alternatives.

© 2001 by CRC Press LLC

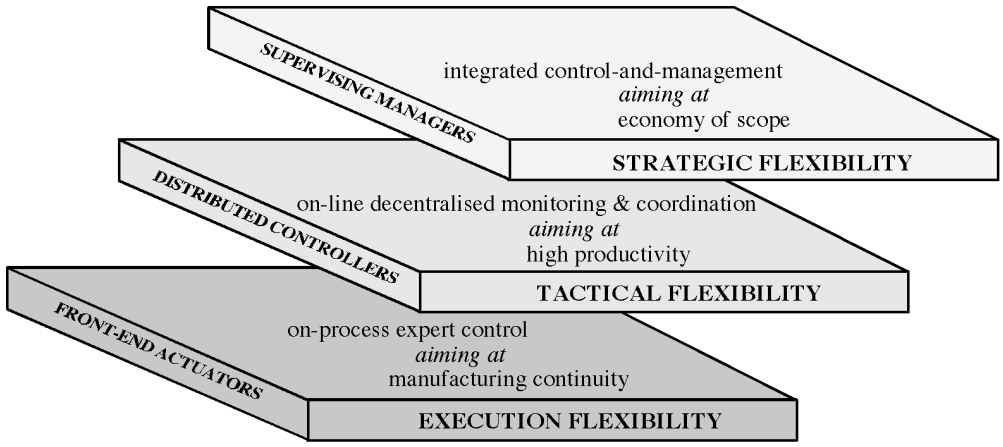


FIGURE 2.11 Knowledge architecture and functional ranges of flexibility.

The Analysis of the Flexibility Factors

The selection of set-ups, properly arranged to grant return on investments, can be inferred by using, as performance index, the flexibility figure, z_f, defined as the number of types changes of product mixes allowed by the functional versatility of the front-end resources, on each (strategic or tactical or, respectively, execution) time span accomplished according to the enterprise policy.

The setting of the assembly equipment was traditionally based on yearly spans. Shorter spans are becoming more relevant in front of market-driven changes; tactical flexibility figures are, presently, evaluated over week-time to shift-time and executional flexibility figures on few seconds to one minute span. The figures are used since initial design stages, for setting and fitting the facility configuration and the related control strategy, according to the decision cycle development recalled in the previous section.

The number of types changes is, necessarily, a previously selected input to establish the physical and the logical resources allocated to the assembly facility. The strategic flexibility figure might, orderly, exploit:

- plant reconfigurability, so that different types of products are assembled after proper specialisation of the functional units and/or modification of the processing layout;
- dispatching updating, with buffers and by-passes, to adapt the assembly scheduling in front of (unexpected or programmed) discontinuities;
- section refixturing, so that the different artefacts are scheduled to be assembled in sequence, by batches, with suitably adapted tools, rigs, and auxiliary equipment;
- agenda-driven, so that workpieces progression involves continuous adaptation of parts feeding and fixtures supplying, since assembly stands are already fit to face the overall product mixes; and
- or further similar setting/fitting actions, granting resources tuning to products.

For the agenda-driven tuning, tactical, and strategic flexibility are balanced. The other situations have a lower tactical flexibility with conditional switching at refixturing or, respectively, at reconfiguration. The executional flexibility figure further depends on the ability of facing emergencies by enabling recovery schedules, thereby assuring to resume the assembly tasks at unexpected events. Several reference time spans are in use, Figure 2.12, depending on the flexibility to be evaluated.

Suitably normal scales are introduced to classify the equipment adaptivity. With due regard to flexible automation assembly facilities, the scale factors of robot-operated cells usually reach low productivity levels as compared to special purpose devices. Higher productivity, however, has to surmount not easily satisfied shop logistics requests, to grant the material dispatching. Example reference data are collected by Figure 2.13. The monthly production, even at the top figure of 300,000 pieces with unattended (three shifts per day) schedules, is far from the performances of fixed automation. Results are further

© 2001 by CRC Press LLC

FLEXIBILITY HORIZONS	DURATION	COMMENTS
execution	fraction of seconds to minutes	this horizon deals with unexpected or planned occurences to restore continuity
tactical	1 workshift = 8 h = 480 min = 28 800 s 1 day = 1, 2 or 3 shifts	this horizon covers current behaviour, granting optimal scheduling planning
strategic	600 000 s (1 shift/day) 1 month = 20.8333 days =1 200 000 s (2 shifts/day) 1 800 000 s (3 shifts/day)	this horizon can be extended to several months, to one (or more than one) year

FIGURE 2.12 Typical time spans of the current flexibility horizons.

PERFORMANCE FIGURE	AVERAGE VALUES
Number of parts to be joined	
- sequencing figure (elemental duty cycle)	6
Front-end manipulation performance	
- unit stroke	5 to 1 s
- duty cycle	30 to 6 s
- delivery (per workshift)	960÷4 800 pieces
Production of a robotic assembly stand	
- standard monthly rate (on a one shift per day rule without refixturing stops)	20 000 ÷100 000 pieces

FIGURE 2.13 Typical reference figures of robot-operated assembly cells.

lowered, when numbers of artefacts type changes are ceaselessly required, needing relevant modifications of the facility equipments, rigs, and fixtures.

The analysis of flexibility requirements, according to enterprise policy, conveniently addresses sets of factors depending on series of hardware/software quantities either through operation contrivances or through planning tricks. The different figures are usefully reported by referring to the net production reached for the assorted mixes of products actually included by the enterprise strategic horizon. The flexibility figure z_f is then characterised by introducing innovation in terms of both the technologies and the methods; the following set of factors is, thereafter, defined:

$$z_f = a_s \;\; a_p \;\; a_d \;\; b_v \;\; b_c \;\; b_s \;\; c_e \;\; c_u \;\; c_d$$

where:

- the three initial factors are useful to make comparisons between consistent results, after proper homogenisation of the enterprise's goals;
  - a_s is the range size figure or dimensional classification of items: from micro devices, to finger or hand sizes; then from easy handling to heavy weight components;

© 2001 by CRC Press LLC

- a_p is the capacity figure or productivity volume of the facility, providing the amount of products to be supplied over the reference time interval; the quantity is suitably replaced by a normalised characteristic;
- a_d is the reciprocal duplication factor, namely the inverse amount of fixtured resources replication that are enabled to work in parallel when needed for assuring the net production by just-in-time schedules; with dedicated automation, each unit assembles different products, with balanced material flows' delivery;

- the subsequent three factors deal with technical operation contrivances, namely:
  - b_v is the versatility figure of the delivered mix, showing the number of product types that might be processed together, on the given time span: year, month, week, day, shift, hour, minute, etc.;
  - b_c is the coordination figure or assembling complexity (involving dextrous fixing, with combined mating motion); auxiliary fixtures and cooperating robots may be requested;
  - b_s is the sequencing figure or number of parts to be joined by elemental movements (these apply, once workpiece and parts are mutually positioned);
- the last three factors are selected with due account of planning tricks accounted for, to obtain reliable delivery programmes:
  - c_e is the fitly assets figure or resources' exploitation ratio, providing the activity time of the allocated facilities with direct involvement in the principal process from the initial (conceptualisation, development, etc.) stage to the current (operation, idle, fixturing, failure, maintenance, etc.) stage;
  - c_u is the uncertainty figure or reciprocal artefacts' design and fabrication robustness to grant conformance to specifications to the (whole) production by means of the principal process alone; and
  - c_d is the delivery margin figure or overproduction referred to trade agreements, so that effectiveness is supported, while completing the delivery within the due date.

The Reference Frames for Selecting Flexibility

The quantities are example reference to understand how to organise cost effective set-ups of the assembly facility for a given application. The nine factors have unlike relevance; suitably scaled estimates are used when a better guess is not affordable. The initial selection of the technological resources moves from functional requirements, stated according to the enterprise policy, by comparing consistent settings. Hints to help specifying these quantities are given in the following.

The size figure is an homogenisation reference; workpieces are, usually, carried by (standardised) pallets modularly selected to accept pieces of different shapes and sizes. Frequently, small pieces are assorted and transferred properly positioned on pallets. The palletised set is considered as a (single) piece, during assembly.

Capacity is fixed as *a priori* estimate, depending on productivity (average assembly time on the unit span, less idle, failure, refixturing, etc. times). If requested volumes outspan the throughput of a single unit, resources paralleling is used. Then equipment can be specialised: individual item takes separate tracks to reduce the assortment allotted to the single unit and to suppress some refixturing stops. The concurrence can also be used for paralleling independent assembly operations.

The (overall) flexibility figures are chosen according to the execution capabilities and are verified as found (ex post) results at the end of the considered time spans.

The versatility figure provides the number of types that are processed by the facility, at the same time, without refixturing:

- on one side, special purpose (high productivity) assembly lines deal with individual items; fixtures availability should even up the product types to be assembled over the considered horizon (with idle resources, waiting for the assigned assembly job); and

© 2001 by CRC Press LLC

- on the opposite side, fully robotised assembly cells can, in principle, operate with self-fixturing mode, making possible to process time varying product mixes avoiding (or minimising) idle resources and resetting stops.

The coordination figure is related to the need of complex assembly tasks, that should be performed by cooperating robots, either:

- simultaneously, by the combined motion of parts and workpiece; or
- sequentially, by changing the posture of workpieces to fulfil the joining cycle.

A careful redesign of the artefacts, usually, can avoid complex assembly tasks.

The sequencing figure represents the number of elemental strokes (giving account of complex joining operations) done by the serial schedule of the assembly duty cycle. Usually ten is an (averaged) maximum and six is a suitably standardised reference. Productivity is improved by enabling multiple assembling by concurrent devices.

The cross-coupling effects of flexible manufacturing reduce the net production and proper planning options need be exploited for choosing a preliminary set-up.

The fitly assets figure measures the portion of the active work-time on the all-duty span; it is used to dress amortisation plans for frozen assets. It is a relative appraisal as compared to nominal productivity; it may be assessed by measuring the duration of actual machine stroke cycle, or more conveniently, of the averaged assembly job over the considered time spans.

The uncertainty figure depends on the quality ranges of the supplied parts and of the delivered artefacts. The production of very cheap products could be compatible with minimising the stroke cycle and increasing the defective parts (to be removed). It shall be noticed that the uncertainty figure differently affects the individual products of any given mix; it depends on monitoring, diagnostics, and recovery implements arranged in advance and on the selected quality tolerances.

The margin figure is a caution introduced as compared to just-in-time delivering for wide products mixes, so that completion of smallest batches is reached in the average without increasing (too much) the stops at refixturing. It leads to over production as compared to some delivery agreements, in order that the completion time is properly inside the due dates for the overall products mix.

Decision Supports and Simulation

The measurement on real plants of the flexibility effects cannot be carried out with generality; limited example cases are, at must, included as validation benchmark. Computer aids are, thus, general reference for the build-up of the basic knowledge and are exploited as consultation support to devise, select, and assess consistent options, in order to fix/reset the CFC frames and to fit/restore the MDM frames. Simulation is the only economical means to gather experimental evidence of advantages and drawbacks that govern-for-flexibility issues, based on integrated control and management, might offer; facilities running conditions are tracked and performance assessed [ACA87b], [ACA87c], [ACA87d], [ACA89a], [ACA96a], [BAL91], [KAN93], [Koj90], [KOJ94], [LIB88], [MIC89], [MIC95b], [MIC96a], [Moi88], [ONO93], [SEL92], [WAR87], [WIE91], with functional models that combine structured descriptions with human-like judgmental abilities.

Expert simulation provides virtual-reality duplication of the plant evolution and visibility on the governing rules. Knowledge coding expands on several layers, Figure 2.14, with oriented scopes, namely:

- the facility description infers the causal relations (structural models) and judgmental frames (behavioural modes);
- the functional modelling leads to generate the algorithmic and the heuristic blocks for the virtual-reality experimentations;
- the testing and evaluation is performed on actual production programmes by varying the governing logic on the strategic, tactical, and executional horizons; and
- the development leads to set the CFC frames and to fit the MDM frames, by iterating decision cycles according to the given area context instanciation.

© 2001 by CRC Press LLC

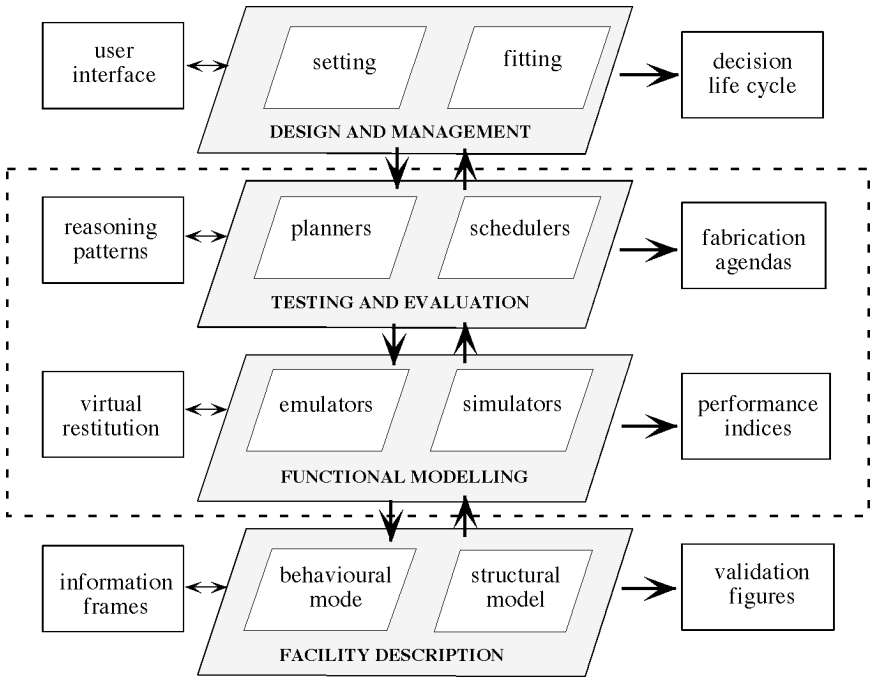


FIGURE 2.14 The knowledge architecture of the emulation/simulation frames.

APPLICATION AREAS	EXPERT MODULES
PLANNING *(at the level, e.g., of plant)*	The software involves classificatory reasoning, singling out the goals, with subclass specification; success depends on the actions satisfying all imposed constraints and is affected by the selection of the employed methods. Bottom-up schedules and backward chaining are worthwhile decision modes for inference, yielding differing implementations, for tracking just-in-time delivery of customers' driven artefacts.
GOVERNING *(at the level, e.g., of facility)*	The software involves integrated production management and control problems; it uses the algorithmic dependencies that correspond to the relations between material entities described by structured knowledge, and exploits the heuristic procedural abilities that correspond to the relationships between the logical entities, provided by judgmental schemata.
DIAGNOSIS *(at the level, e.g., of equipment)*	The software is based on the acknowledgement of "facts", by performing the classification of information expressed as "attribute, value and belief"-elements; success is affected by search efficiency. Condition-monitoring maintenance is an option to grant zero-defects production by preserving process properties.

FIGURE 2.15 Emulation/simulation environments and example application areas.

The software includes two series of modules: the first generates the facility dynamics (structural frame); the second provides the judgmental logics (behavioural frame). The package assures the testing on alternative set-ups by simulation; the engineer exploits the option at the development stage provided that functional models are established on parametrical bases. The govern modules supply the means to evaluate flexibility effects, when plans are enabled by decentralised control and supervisory management, as case arises, along the strategic, tactical, or execution spans. Different goals, moreover, require different expert codes, to accomplish, e.g., the planning, diagnosis, govern, etc. tasks, Figure 2.15, each time based on programming aids that commonly share knowledge-based patterns for the easy coding of the decision logic and the emulation of human expertise.

© 2001 by CRC Press LLC

The knowledge-based codes have been developed as case application of AI methods in production engineering. The cases here considered characterise by two facts:

- multi-agent development philosophy is followed: attention is focused on shop floor activity (the assembly process), assuming cooperating evolution of frontend agents, within a message passing communication environment; the decision logic should be effective for the real-time govern of the concurrent joining schedules; and
- functional characterisation approach is exploited: expert simulation is used aiming at prototyping for evolution by iterating evaluation-modification cycles; the decision logic emulates the reasoning patterns of area experts, to obtain the performance rank of each tested alternative.

The basic features of the emulation/simulation software aids are summarised in the next point; the subsequent point gives indications on its current utilisation.

The Emulation/Simulation Software Aids

Knowledge-based systems are a useful implement for engineering applications based on empirical contexts. The ability of encoding information by knowledge, rather than plain data, is basic motivation. Knowledge is information with attached relational context. In the software, one distinguishes, Figure 2.16:

- declarative knowledge, typically: structural attributes of the assembly resources, at the input interface; and: process information, at the output interface; updated information is stored by the current memory and makes possible the procedural instanciation; and
- procedural knowledge, typically, the behavioural properties of the assembly facilities; rules are common coding to express the methods (how to accomplish actions or take decisions) for processing the declarative knowledge.

Object-oriented languages are good options. Objects, with attribute and method coding, grant a unifying programming aid; the related relational frame supports inheritance and makes it possible to define generic elements, reusable as case arises; the structure induces incremental descriptions and classification.

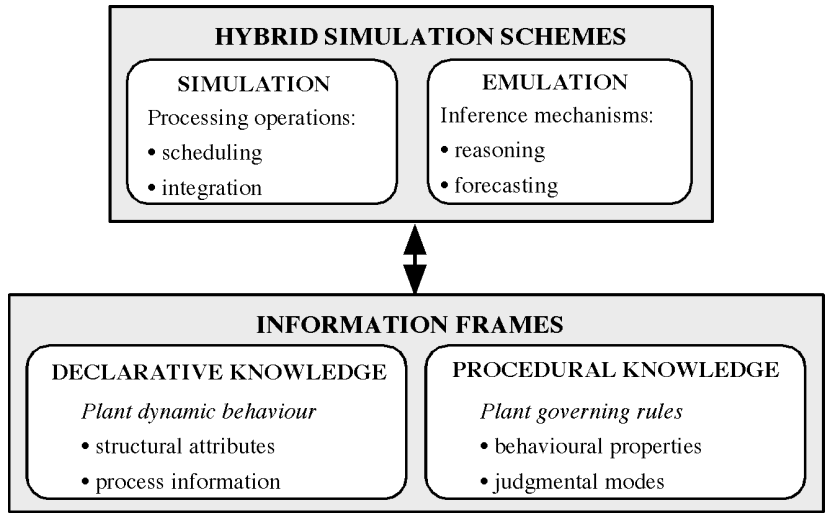


FIGURE 2.16 Typical arrangement of a knowledge-based simulator.

© 2001 by CRC Press LLC

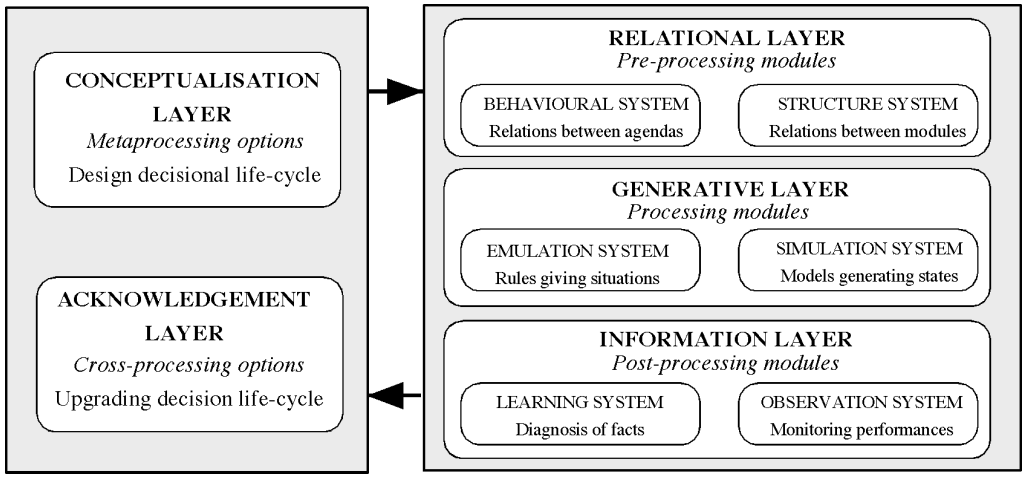


FIGURE 2.17 Multi-layer architecture of codes for intelligent manufacturing.

The philosophy of knowledge-based codes for intelligent manufacturing is shown by Figure 2.17. The software develops as a multi-layer construction, with vertical connections among the relational, the generative, and the information layers. The relational layer embeds preprocessing modules, as friendly interface for model selection and agenda setting, to help defining the structural attributes of the assembly facilities and the behavioural properties of the related governing logic. The availability of integrated simulation environments is obtained by referring to specialised data-bases connected through management facilities with both the generative and the information layers so that virtual-reality work sessions are run with full transparency of the control flow. The information layer performs restitution operations through post processing modules. The user can call for graphic restitutions; the process information is shown as sequences of relevant facts with situational specifications, or is processed to provide performance evaluation as compared to competing schedules.

The generative layer contains the solving capabilities; it propagates causal responses by means of algorithmic blocks and acknowledges consistent suppositions through heuristic blocks. The decision cycle, aiming at achieving flexible automation concepts, is heavily conditioned by the iteration of the emulation/simulation loops.

Figure 2.17 further shows a conceptualisation and an acknowledgment layer used as interfacing options to an (outer) learning loop closed by an intelligent governor responsible to adapt control and management to the changing enterprise policy:

- at the conceptualisation layer, the user has access to the structural models and to the related behavioural modes; automatic exploitation of the technological versatility can be devised to enable cooperative and coordinated actions; the meta-processing abilities grant specification transparency and fore-knowledge opportunity to the (subsequent) relational layer;
- the acknowledgement layer supplies process-data visibility; the assessed facts are used to update resources describing data and to initialise provisional analyses on the chosen flexibility horizon; at this layer, specialised cross-processors, by automatic accrediting the hypothesised justification knowledge, might enable the expert-simulator to operate on-line, for adaptive assembly.

© 2001 by CRC Press LLC

Testing the Achievements of Flexible Assembling

The correct setting of the assembly facilities is assessed by virtual reality simulation with due regard of the enterprise policy and manufacturing strategies. Reconfigurable layouts based on modular units are typical reference to maximise productivity; robotised solutions are prised when one-of-a-kind production should be granted. The opportunities are hereafter commented in relation to the flexibility goals.

Mass production, aiming at maximising productivity by the economy of scale, being characterised by a vanishing flexibility figure z_f has the design stage already fulfilled at the conceptualisation and acknowledgement layers and does not require the relational, information, or generative layers. On the reciprocal side, the one-of-a-kind production characterises by the largest flexibility figure. In between, the figure takes intermediate values with higher placing obtained by alternative means, namely:

- to modify the assembly set-up, with off-process operations, aiming at functional resources granting front-end equipment specifically suited to the scheduled products; the compression of time-to-market is obtained by simultaneous engineering through balanced development of product and process; and
- to expand the technological versatility of the assembly facilities; this issue is reached by using robotic equipment having suitable operational dexterity and functional autonomy, and multiplying the processing units, so that, by proper shop logistics, the material flows are addressed to the convenient special purpose workstations.

The simulation is used, on these premises, to select the technological resources, to set the assembly layout and to fit the decision manifold. For factory automation, the govern blocks are directly interfaced to the facilities, with twofold operativity:

- to generate updated information for reset and outfit operations, whenever the product mix changes and a different production capacity should be established; and
- to enable joint management and control, so that recovery options are exploited at emergencies and optimal schedules are preserved along the (steady) tactical horizons.

Inclusion of inference capabilities makes possible anticipatory management loops for responses that close as standard govern-for-flexibility property for the on-process adaptive control of the assembly cycles. The option is central from two standpoints:

- for assessment purposes (as a CAD option), to expand the expertise of production engineers, allowing for interactive enhancement of knowledge references; and
- for implementation purposes (as a CAM option), to grant flexibility to the assembly, within the considered range, providing transparent adaptivity to the control.

Shop floor control and management means on-line inference capabilities; the set-up needs be standardised to be easily adapted to varying functional requests or operation environments. The expert governor module is, then, connected on-process to work in real time to control the assembly unit and to manage the assembly plans. The typical components of the expert module are:

- the conditioning logic, encoding the procedural knowledge;
- the data-memory, embedding updated declarative knowledge with current data; and
- the inference engine, activating the decision cycle by information-match, conflict-resolution, and rules-firing.

Choice of management operations exploits learning loops aiming at assessing the cross-coupling effects of flexibility on efficiency; these effects are related to the control operations selected for the facility production capacity, in view of the delivery schedules and due dates. Efficiency is evaluated after collection of sufficient data by repeated simulation tests. The upshots of the govern-for-flexibility of the shop floor operations are acknowledged, for upgrading the assembly efficiency, fully enabling the embedded technological versatility with due account of leanness. For capacity allocation, the control and management

© 2001 by CRC Press LLC

of flexibility is made up of decision options in connection, e.g., with shop floor logistic, operation cycles schedules, production agendas planning, and capacity requirement setting. Each option offers several alternatives regarding how assembly capacity can be allocated to meet requests exploiting simultaneous engineering practice of adapting products and processes by stepwise betterment.

Example Developments

The following two sections present application examples that distinguish because of product quality and production volumes. Inexpensive artefacts require comparably cheap manufacturing facilities. When rentability has to be reached by mass delivery, assembly has to be done with special purpose equipments, by high productivity layouts; trade competitiveness may lead to run for abrupt changes to follow fashion or personal whims. Modular facilities happen to be relevant options; they are reused, combining the modules into new set ups each time different products are scheduled. Flexibility is managed off-process through re-configurability; products and processes are developed according to simultaneous engineering paradigms reducing the time-to-market since the basic (assembly) functional modules are already available.

The first example deals with the fabrication, by batches of million, of small artefacts delivered, as specialised componentry of the house electrical wiring, for personalising individual surroundings. The design is undertaken with the package MAR-SIFIP, based on the Taylor software (F&H). The solution is sought by redesigning the assembly section so that parts can be modified according to fashion trends, then produced with the same resources; these are conceived as modular functional units for elemental operations (click joining, spring locking, screw linking, tail cutting, glue sticking, ultrasonic soldering, etc.). Facility effectiveness is investigated off-process, modifying task sequencing and coordination according to the simulation results obtained with the package MAR-SIFIP. Productivity is, thereafter, optimised with the assembling actually performed by special purpose equipment.

As second example, a similar manufacturing situation is considered in view of extra opportunities offered by modularity joined to some adaptivity provided by dispatching redundancy. The study is performed by means of the code MAA-SIFIP based on the MODSIM III package (CACI). The assembly line is, this time, assumed to follow a closed layout, with branching tracks and side lanes; the workpieces have accumulation buffers, with a transfer supervisor managing priorities, overtakings, recyclings, etc. In the example case, redundancies exploit local carousels, with small extra investments; on-process limited flexibility assures some recovery abilities, with reasonable downgrading. The merging of modularisation of the basic assembling functions and flexible dispatching of the material supplying looks to be a promising option; construction of the modular units could face technology limits; resetting time and adaptations need be evaluated in terms of economical returns as compared to the higher complexity innovations, more based on computer implements.

When rentability has to be reached by customised delivery, assembly could be done by robotic equipment by high flexibility layouts. Expert governors become the winning option; they assure integrated control and management capabilities, with on-line and on-process flexibility, properly enabled on the strategic, tactical, or execution horizons. Products and processes are continuously adapted, based on decision cycles, that avail of the results provided be emulation/simulation of the real facilities, by comparatively sophisticated software. Section 2.5 of this chapter presents two more studies showing the flexibility behaviour provided by robotised facilities.

The third example, thus, considers the setting of a robotic assembly cell of a flexible specialisation firm, that offers itself as active supplier by cooperating with the end manufacturer for the design of customised artefacts. The delivery is concerned by high quality parts purposely redesigned before inclusion in the final product. Simultaneous engineering should be enabled for the joint development of parts and process to minimise the time to market. The implementation is supported by the code RAD-SIFIP built with the ROANS package (JHF). The development tool provides a supervisor for fitting the assembly sequences and for setting the work and test schedules. The design aimed at using standard resources (two SCARA robots, vibration feeders, regular fixtures, and instruments, etc.), supervised by a block supporting virtual equipment generation. The switching to new orders exploits the code for virtual reality

© 2001 by CRC Press LLC

experimentation of the control set-ups; productivity of the robotised assembly solution is then transferred on-process and controlled over the scheduled tactical horizon (totally unattended shifts are included) by means of a supervisor whose simple function is confined to restore the programmed task sequences.

The fourth case is concerned by fitting govern-for-flexibility options for an assembly section which has to face varying product mixes. The design is done using the package XAS-SIFIP developed with the expert shell G2 (GENSYM). Several cells work in parallel, with feeding actions (pick, transfer, place) governed by heuristic rules to maximise the delivered output and minimise the work-in-progress. During simulation, failure, and repair occurrences are statistically generated to test recovery flexibility options. The fabrication agendas are provided and (through virtual reality) the production plans are fitted to approach just-in-time policies and to obtain optimal schedules on the tactical horizons. The overall assessment of the enterprise policy is investigated on the strategic horizons comparing alternative decision schemes.

The cost effectiveness for the different examples is concerned by the methods of simultaneous engineering, through the iteration of decision cycles design-feedback-choice and setting-testing-fitting closed for product and process. Improvements are related to innovations in computer-integrated manufacturing and engineering (CIME), such as cooperative knowledge processing; the issue is, perhaps, seldom prised for engineers' concern. However, production engineering in these past years seems to be a fast moving field for AI developments and a good deal of tools are entering into practice to enable economy of scope. The case studies, hereafter presented, refer to real applications gradually enabled with tests on the technology appropriateness and careful checks on the leanness of the proposed facility. The developments are given with stress on the methodological aspects more than on the specific results (these are suitably rearranged to preserve the confidentiality of the individual business enterprise situation).

2.4 Reconfigurable Set-Ups Assembly Facilities

Mass production reaches effectiveness and reliability by product oriented automation. The use of purposely developed devices requires important investments each time the enterprise policy looks for different artefacts; the switching, with short time-to-market, to new products needs specify how to make available the pertinent processing equipment. A chance is given by modular layouts obtained by combining functional blocks into high effective facilities on the time horizon of the planned order and by recombining the same blocks into alternative set-ups each time the case arises. The plant reset logic is enabled as the order of new artefacts is initialised; product and process are jointly developed according to simultaneous engineering paradigms.

Versatility is achieved by layout updatings; the setting is concerned by the structural frames, CFC: **c**omponents, **f**acility configuration, and **c**ontrol. Modular assembly facilities then reach fixed automation performance while offering operation adaptivity due to the off-process resetting abilities. By simulation, the modular facilities are shown to preserve highest productivity provided that the driving disturbances (unexpected occurrences, failures, etc.) are mastered to remain within the dedicated solutions ranges.

Modularity is consistent with front-end settings based on transfer assembly lines, in fact units are added or removed quite simply, local equipment (vibratory feeders, etc.) can be included or specialised with noteworthy freedom, workstations may be expanded into separate units, splitting a given assembly cycle into subtasks, loading/unloading fixtures can be selected to supply convenient versatility for the assembly of the considered products, etc. The selection of the basic modules bears wide freedom on condition that the whole assembling is arranged to be performed with a given order. The requirement, depending on the application, may happen to be too exacting; adaptivity should be brought further into shop logistic introducing workpieces route changes with side ways and/or return loops between workstations by respect to the straight assembly line. This second arrangement is consistent with on-process flexibility, through re-route planning.

In any case, the functional models presume detailed specification of process primitives to make the hardware and software equipment correspond to the relational frames of the relevant material and decisional transformations. The software, purposely developed for each case application, leads to simulation packages

© 2001 by CRC Press LLC

that profitably make use of oriented languages to bring mapping up to the level of the knowledge representation. The related modular hardware is being conceived on a two kinds status:

- application-driven modules for processing, testing, etc. and, in general, for the particular operations related to each special case development; and
- service supporting modules, for shop logistics, planning, etc. and, in general, for every auxiliary action required by the assembly process.

The modules of the first kind directly depend on the assembling process to be fulfilled. The functional characterisation of task primitives follows lines akin to the setting of special purpose devices with modifications to propagate easy interconnection and to supply reconfigurability. The careful definition of the assembly functions is important, to obtain reuse of the facility over an extended mix of items, making possible to develop new artefacts while preserving the processing primitives. As for the modules of the second kind, specialised suppliers offer different sets for the builup of the workpieces transfer tracks, transport actuation units, pallet travelling trays, assembly posture stands, parts feeding devices, etc., each application case profits by such standard modules.

This study aims at supporting the computer aids allowing the through-out assessment of the effects that fit-out options might cause bringing forth the innovation of modular layouts. The reset and the outfit decision logic is assessed and confirmed with aids based on the measurement of plant performance and the comparison of current figures against the expected levels of efficiency. To offer this decision aids, functional modelling and expert simulation of the manufacturing processes, providing virtual reality display of actual production plans are used as main reference techniques.

A setting is really cost effective only if the number of independent modules is suitably bounded and their reuse is fully granted. Modularity, moreover, suggests additional specific opportunities, namely:

- the redesign of products or processes is easily transferred by recombining the modules; or, reciprocally, convenient set-ups give hints for upgrading the artefacts design;
- the functional specialisation characterises each module; trend-monitoring maintenance organisation strictly provides a distributed diagnostical environment;
- the inclusion of extra modules can be planned at critical workstations; conversely, the critical tasks can be rescheduled by minimising the risk of spoiling the added value; and
- the task segmentation makes it possible to include robots instead of special purpose units exploiting modularity for a smooth approach to flexible automation.

The section will tackle over the modular assembling perspective as follows:

- the assembly problem of low price mass production artefacts is considered and issues for modifying the production with time-to-market constraints are discussed;
- a first solution, based on full modularisation of an assembly transfer line is presented and the software aids (for, both, off-process and on-process support) are stated;
- a second solution, based on modularisation of the functional units joined to an adaptive shop logistic setting, is discussed in view of the related real-time flexibility;
- the solutions are acknowledged in terms of the versatility requirements to be forecast off-process through re-configurability, with account of the productivity figures; and
- the developments are summarised, by referring to case example applications and results, as issues of the prospected approach.

Modular Assembly Transfer Lines

The first application deals with a facility engaged to assemble sets of small electrical appliances (plugs, switches, gauges, hook-ups, etc.) to complement the house interior wiring. The overall size of each device is about 30 to 150 mm, with 5 to 15 parts to be joined for each typical assembly cycle. They are delivered

© 2001 by CRC Press LLC

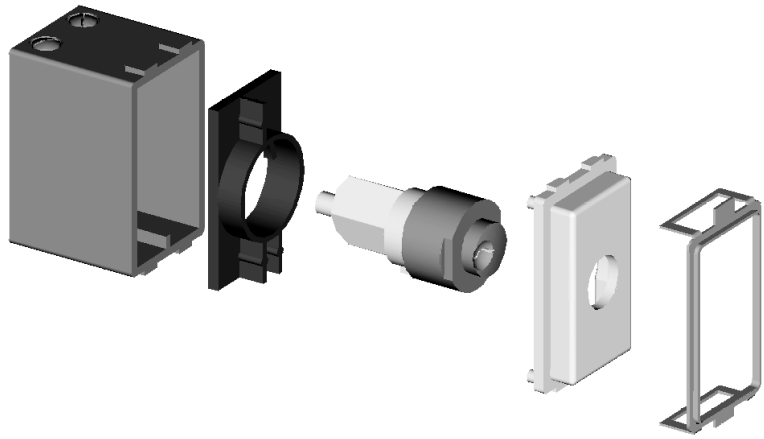


FIGURE 2.18 Exploded view of a typical item (key-switch) for house-wiring.

as outfits of comparatively expensive finishing craftworks, that distinguish each other, to customise the final setting and are subject to modifications as technology, standards, or fashion change. Devices are delivered by large batches; manufacturer's return depends on the fabrication rate and requires very high productivity upto now reached only by special purpose automatic units. The manufacturing of such harness, first of all, requires good quality molds to obtain accurate parts; the assembled devices need, finally, get over suitable quality tests before delivering. As soon as molds are available, production can address the new parts to be joined; quick resetting of the modular assembly facility and the befittingness of the schedules are, thereafter, winning options.

A sample key-switch is shown in Figure 2.18. The device is based on three different kinds of subassemblies, separately produced, and fed to the final modular assembly facility. The study starts by the redesign of, both, the desired device and the assembling process, so that the suitably adapted componentry could be joined reusing the same modules. This requires to singularise each job to have independent execution, to refer to a functional unit for each elemental operation, and set the duty cycle according to the artefact's joining sequence.

Functional Characterisation of Modular Assembling Facilities

Functional bent and resort (not device orientation) are inherent merit of modularity. The following two classes of modules have, in general, to be considered task-driven (first kind) modules, namely:

- positioning modules, including special purpose pallets, fixtures, sensors, etc.;
- processing modules, such as: screwdrivers, joining fixtures, clinching rigs etc.;
- testing modules, for supervision, quality control, and monitoring duties; and
- govern modules, to generate and perform task-adaptive control;

service-supporting (second kind) modules, such as:

- transfer modules, including general purpose instrumental equipment;
- shop-logistic modules, for buffering, transportation, and handling functions;
- parts feeding modules, to enable the local storing and sorting dispatching; and
- fitting modules, to assure compatibility through standard interconnections.

The segmentation into classes is done somehow arbitrarily depending on subkinds of modules strictly required by the application driven development. As a all, every physical and logical resource needed for fulfilling the assembly cycles is specified to support reconfigurability. Adaptivity aims at establishing the

© 2001 by CRC Press LLC

highest productivity figures, by the redesign of the process set-up simultaneously with the new product, through a multiple step approach, based on the:

- recognition of the manufacturing process (for consistent quality/efficiency setting);
- choice of a strategic enterprise organisation (to expand market-share figures);
- reset of the production resources (to enable proper executional effectiveness);
- comparison of tactical alternatives (to optimise the return on investments); and
- balancing products and processes design requirements (for highest robustness).

The assembling final set-ups characterise by several alternatives. Simulation provides the reference data for the progression of the project, e.g., to assess the effects of:

- addition of assembly units, after splitting a duty cycle into elemental actions;
- modification of the assembly sequence, with ordering index based on job difficulty;
- updating of the schedules with change of the priorities setting; and
- the likes.

The study moved, according to the outlined approach, specifying:

- processing units (to perform parts mating and joining tasks); handling fixtures from and to feeding stations; auxiliary units, to check parts codes and postures, to supervise the assembly cycle, to remove (possibly) defective components, to monitor finished items quality (impedance tests, components integrity, etc.); govern units, to program the work synchronisation, sequencing, coordination, etc. and to grant the achievement of the productivity figures. Example characterising features of the main application-driven modules are as follows:
- the processing units are required to accomplish tasks, such as click joining (or spring locking), screw linking, tails cutting, glue sticking, and ultrasonic soldering; and
- the testing units are charged to perform tasks, such as inspecting parts and artefacts integrity, verifying the geometric tolerances and mechanical strength, and checking the electrical insulation.

Each task is undertaken by a specialised unit purposely designed in view of the selected function. The last task, for instance, is planned for quality control. The first refers to the axial thrust assembly of parts with shaped rims to help plastic (elastic) inserting. The individual units are easily reused, if proper shrewdness is applied in the design of the new componentry.

Special effort aims at standardising the service modules, namely, in particular the local handling rigs based on a pallet carousel around the assembly facility, and the section transport service, feeding (on duty) workstations. The choice of the shop transfer system is important to make reconfigurability easy while keeping very high production rates. A ring slide-ways layout has been retained since addition, removal, or spacing of functional modules is done by replacing pieces of linear rail tracks. Supply of items can be performed with continuity, starting with half finished workpieces, to end with the artefact, after completion (parts addition, fixing, testing, etc.).

The plant is set up by selecting a proper set of functional units and combining the related assembly stands; these are equipped with the usual parts feeding and orienting devices and sometimes include positioning and handling equipment. The parts are automatically loaded on pallets and mounted during the assembly process and the items are eventually taken off at the end of the cycle while the carriers are recycled; complex subassemblies are sometimes asynchronously provided during production and are available at fixed assembly stands by means of belt conveyors or roller chains. Both kinds of transport systems (ring slideways and belt/chain conveyors) also act as local buffers providing interoperational storing capabilities or subassemblies stocking.

The stations are fed by the ring slide-way and the following elemental actions are repeated along the tracks: insertion of screws or other items in plates of different sizes, screws fastening, withdrawal of subassemblies, recognition and positioning of parts/sub-assemblies, check of parts presence, fastening of safety clips, and functional testing.

© 2001 by CRC Press LLC

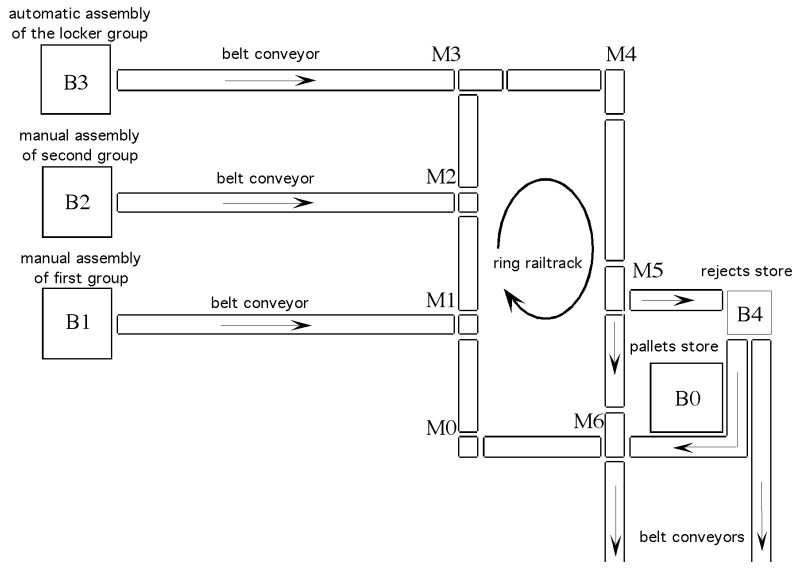


FIGURE 2.19 Reference layout of the modular assembly line.

The Assessment of the Assembly Cycles by Simulation

The example device of this first sample case, a key-operated safety switch, is based on three different subassemblies separately produced and fed to the modular assembling facility through buffers B_1, B_2, and B_3, Figure 2.19. In the present example, two different types of (internal) contact groups are manually assembled at stands B_1 and B_2, while the locker-group is automatically joined aside and small batches of four items are discontinuously loaded onto buffer B_3; these elements are then carried by the belt conveyors to the stands M_1, M_2, and M_3, where they are assembled with the switch body.

Standard pallets are used to carry the key-switch bodies to the assembly stands; initially they are stored at the central buffer B_0 and, when production starts, they are moved to the station M_0 by means of the ring railtrack. Here, the switch bodies are automatically fastened to the pallet and minor parts are put on place. Items, then, move to the next station (M_1) where they are joined to the first subassembly; the process is repeated at stations M_2 and M_3, as already described; at station M_4, the assembly is finished, by adding undercover and cover, that are fastened by a clip. At the next station (M_5), products are inspected to assess the quality of the overall process: defective parts are rejected and go, with their pallet, to the buffer B_4. As soon as four rejects have accumulated, they are turned into scrap and pallets, after manual disassembly, are again moved back to the central store to become available to start a new assembly cycle. Approved products are directly taken off the carrier, after inspection at the M_5 station; their pallets, as well, are free again to resume a new cycle. Figure 2.20 shows the visual display of the layout offered by the simulation code; on the lower part of the screen dynamic icons show relevant data on evolving production: throughput (ready parts per hour), level of scrap in buffer B_4, number of available carriers, and overall utilisation coefficient of machinery.

Because of the irregular supply of subassemblies and the stochastic rejection of parts, the number of empty pallets varies during the work-shift, as well as the number of rejects waiting for manual unloading. Several simulation tests have been carried on to assess the machines utilisation ratios and to find the more convenient number of pallets. The impact of carriers' translation velocity has been studied, too, and series of simulation runs have been performed to test different alternatives: some resulting data are gathered in Figure 2.21 and 2.22.

© 2001 by CRC Press LLC

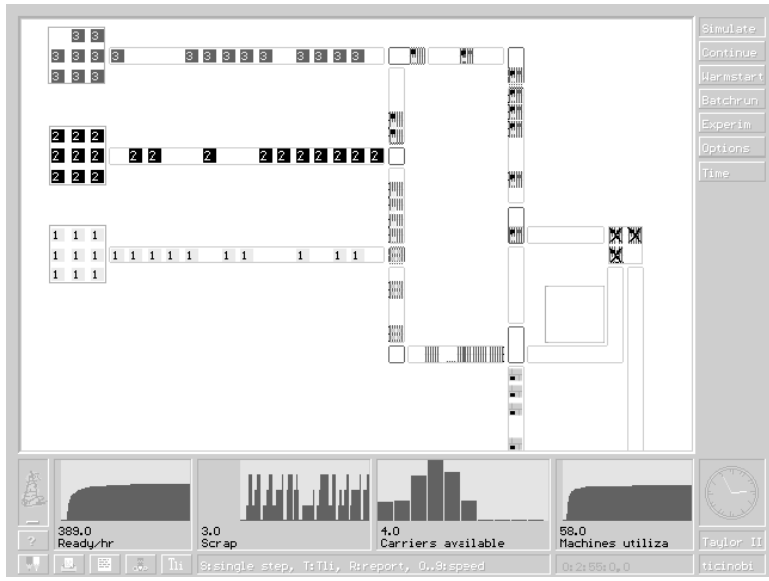


FIGURE 2.20 Example outputs of the MAR-SIFIP simulation code.

	v = 0.2 m/s		v = 0.3 m/s		v = 0.6 m/s	
12 pallets	158	23	171	21	327	11
24 pallets	221	16	378	10	617	6
36 pallets	411	8	570	6	591	6
48 pallets	434	8	603	6	627	6
60 pallets	387	9	334	11	614	6

FIGURE 2.21 Production rate (part/h, light background) and throughput (s/prt, dark background).

The analysis of data shows that, depending upon the maximum allowed speed of the transfer system, the number of carriers approaches rapidly a limit value beyond which no benefit is foreseen for the production with the risk of queues and jams along the railtracks. For instance, if the pallet speed is raised as high as 0.6 m/s, the throughput time per assembled item reaches the limit value of about 6 s with 24 carriers only; further improvements are hard to be achieved since the longest operation takes 5.5 seconds. Figure 2.23 shows the bar charts of the production figures for the considered case.

The practical evaluation of the effectiveness of the selected modular assembly facility has been assessed by means of computer simulation. The code MAR-SIFIP, based on the Taylor package has been developed. The basic configuration includes data about physical resources (processing and transfer units, parts, and pallets) and data on logic resources (monitoring, decision, or command units). The modules are detailed, modified, and instanciated to suit the case situations, and the resulting facility layout is reset with help of the animation interface for optimal efficiency.

Taylor II Simulation Software by F&H is a tool supplied to model, simulate, animate, and analyse processes in manufacturing and material handling. The code can be exploited to support investment decision, to verify design of manufacturing plants, to experiment with different planning strategies, or

© 2001 by CRC Press LLC

	v = 0.2 m/s	v = 0.3 m/s	v = 0.6 m/s
12 pallets	19	28	48
24 pallets	34	59	91
36 pallets	64	86	91
48 pallets	70	90	92
60 pallets	66	50	91

FIGURE 2.22 Overall equipment utilisation ratio (percent).

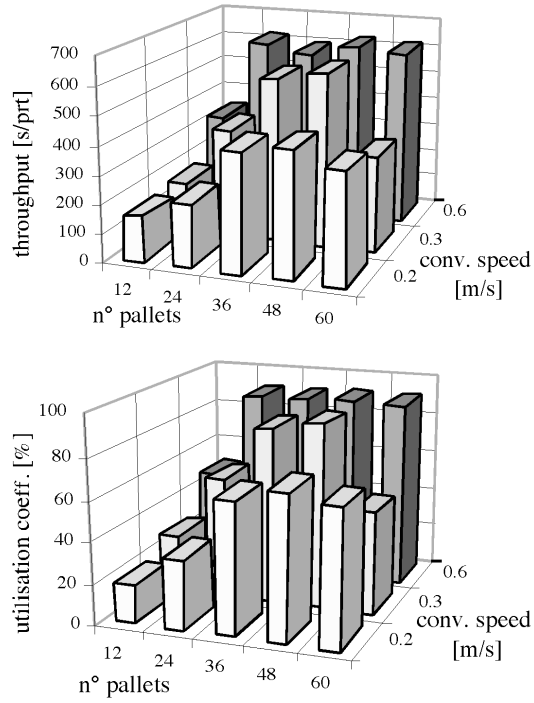


FIGURE 2.23 Production figures and resources utilisation ratios.

to test performance offered by competing material handling installations. It will show the throughput of a production facility, identify bottlenecks, measure lead-times, and report utilisation of resources.

The package shares the principal features of the many already available tools, namely the building of models is pretty fast and easy, the primitive constructs allow the user to model various levels of complexity, the models can either be quite abstract and conceptual or quite detailed and technical, the execution speed of simulation is rather fast, the simulation models are animated immediately (also in 3D), without any extra work, the user is helped with analysis through reports and graphs, built in statistical features help the user with pre- and postsimulation analyses, and the package is open and users can communicate with external programs (e.g., spreadsheets). The program has been developed by means of five fundamental entities: elements, jobs, routings, stages, and products.

The most relevant elements that have been used in the example are the workstations, the buffers, and the conveyors, and they are in charge of the decentralised governing logic. The stations (i.e., assembly stands) are mainly characterised by the capacity (1 pallet per time) and by the input queue discipline (FIFO); the components and spare material are figured to be always available, but for the joining of subassemblies the management of synchronised appointments was needed. The buffer's main feature is

© 2001 by CRC Press LLC

capacity, but also queue discipline (FIFO in this case) and entry/exit conditions may be important; the I/O buffers are special elements that are used, in this model, only to create the subassemblies and put them into the assembly plant. The conveyors, moreover, are of two different type in the present model and are characterised by their speed, length, spacing requirements, capacity, etc. The jobs are used to model the operations that can take place on single elements and describe how an operation is done in terms of cycle time, batch size, priority, waste, etc. In this model cycle times are usually fixed and stochastic distributions are used only for human-related operations (i.e., the production of two subassemblies) while the priority has not been used in the present example, having only one item production. The routings, that are composed of a number of stages, define which job is used by the single stage and what the next stage is: alternate routings can be easily managed, depending on ongoing production, machine status, work-in-progress, queue length, etc. The products, finally, flow through the model and an initial stock is defined at every stage: every product has its own name, graphic icon weight, cost, etc.; moreover, it is separately monitored during simulation and accounted for by final statistics.

A specific interface language is used to write the triggers that rule the simulation and generate the assembly process. An independent entry provides the statistically generated interrupts that give account of (possible) failures. The objects corresponding to functional or transfer units have fields for the MTBF (mean time between failures) and for the MTTR (mean time to repair) data. Items are endowed by a quality binary label, updated according to the reliableness of the performed action: when the label is switched off, the workpiece is removed from the assembly ring. It should be noted that example artefacts, delivered by huge batches, are comparatively cheap; they shell undergo functional and aesthetic checks and defective pieces are totally rejected, since shipping is done only for fully accepted products.

Betterments Suggested by the Modularity of the Assembly Units

To develop a modular facility, extensive simulation has to be performed with the MAR-SIFIP program before validation of the proper assembly set-up. Reference aspects are used to characterise each physical resource, e.g., processing units: functional properties, ordering sequencing, etc.; ring railway: number of pallets, processing complexity, etc.; transfer/buffering belt: queue length, pick-and-place service, etc. Several simulation work sessions should be performed to assess the effectiveness ranges of a given assembling schedule. The checks have to cover:

- unbalance bounds: bottlenecks in a line with a station having longer duty cycle;
- reliability figures: productivity with units characterised by different MTTR or MTBF data;
- quality control: effectiveness of agendas based on operation order sequencing;
- setting alternatives: operation/functional attributes selected for the local workstations;
- artefacts redesign: properties modification by design for assembly rules; and
- product/process integration: simultaneous engineering betterment cycles.

To reach the enterprise's shipment strategy, fast cadencies are needed for each unit so that the slightest trouble quickly creates unbalances. Parts quality falls off on the final products characterisation and customer's satisfaction cannot risk the delivery of partially defective lots. The highest productivity is obviously reached by modular arrangements of stations with exactly the same time cycles, when the manufacturing progresses with zero defects and without resources failures. But: zero defects means sufficient tolerances by respect to customer's satisfaction; resources full availability means sufficient redundancy with set apart modules that authorise reliability by beforehand maintenance.

Return on investment suggests to check every useful alternative. The case application particularly suits the innovation of modularity: analysis of the assembly sequences provides sharp definition of the processing units; and consideration of logistics brings to well assessed transfer and handling devices. The different modules are conceived and built with investments comparable with fixed automation rigs. On such premises, the study looked for practical extensions directly related to the ability of making transparent mapping of duty cycles into modular set-ups. Simultaneous product and process design, then, can be further explored, aiming at simple, nevertheless worthy, tricks to improve the assembling efficiency such as

© 2001 by CRC Press LLC

- to anticipate, in the work cycle, the operations with highest failure risks so that possibly defective pieces are immediately removed suppressing further processing costs;
- to set upstream the workstations with the shortest processing time and highest reliability figures to grant safe supply of downstream stations;
- to investigate the feasibility of resource duplication or of functional redundancy, starting from critical upstream stations; the trick shall exploit monitoring of each manufacturing device for trend diagnostics, so to enable recovery flexibility by the adaptive fitting of the production schedules.

Based on simulation results, the duty cycles can differently be stated by adapting layout and functional modules; sequencing and coordination are granted by standardising also the interfacing with the transferring modules, organised as a line discontinued by stops for performing the scheduled actions. During simulation, the task adaptivity is ruled by a supervising module incorporating heuristics chaining and judgmental checks. The expert simulator in this case is used off-process to assess the best plant layout at each new order; the acknowledged scheduling is then transferred to work on-line directly operating with the physical resources. Virtual reality exploration, to sum up, leads to exploit modularity for arranging optimal layouts based on fully beforehand assigned constraints.

Unbalances, however, could arise due to failures (of functional units, feeding devices, etc.) appearing, on-process, as unexpected occurrences. Then, the conveyor supply may create bottlenecks with queues up to cancellation of upstream feeding and supply stoppage to downstream stations. The aspect could be counterbalanced only partially for assembling processes strictly based on transfer-lines; statistical assessment of the average disturbances has to be considered and inter-operational coveyor buffering capability should be set such as to preserve conservative recovery figures while facing the discontinuities.

Once the modular facility is running, the simulation code MAR-SIFIP becomes standard development aid for the choice of the convenient set-up matching the newly designed products. The project has been addressed also to assess the smooth entry of the modular arrangement. Virtual reality results have been collected to compare the following issues:

- the simultaneous design and setting of equivalent artefacts (and fabrication agenda) and related manufacturing facility; modularity is exploited to reach higher efficiency, by resetting the assembly section configuration and by redesigning the products, with the options such as the above recalled tricks; and
- the design and setting of a reconfigurable (auxiliary) manufacturing line, that might operate in parallel with (existing) fixed-automation lines, to supply flexibility by facing order peaks or by damping away transients during redesign, development, and fixturing of new product oriented facilities.

The implementation of this (auxiliary) modular facility is related to the capability of managing the (tactical) flexibility about the steady (strategic) ranges still provided by (sets of) product oriented fixed automation facilities. Tactical horizons cover, at occurrences, the approximately month span; the modular facility helps the transition to the new fixed automation set-ups by assuring preliminary delivering each time new artefacts are requested. The auxiliary assembly section extends the enterprise adaptivity supporting customers' satisfaction with minimal time-to-market. Similarly, the experimentation on different design-for-assembly issues or on alternative assembling sequences can compare the effectiveness of the related set-up, providing criteria for choosing the appropriate configuration of the special purpose facilities once experience on preliminary delivering has provided sufficient information on how choices should be addressed.

Modularity of Assembly Lines with Buffers and By-Passes

Modularity associated with assembly lines is a relevant issue to preserve productivity, while supporting reconfigurability with reuse of the functional units. The main drawback is represented by the strict sequentiality that hinders time balancing for the processing at a time, of more than one item, with different

© 2001 by CRC Press LLC

length duty cycles. Situations even grow worse when the facility suffers from resources' unavailability or failure. A careful simulation study can possibly state, by off-process setting, the equalisation of the work station duty cycles on the assembly line for a single artefact; the issue cannot be extended to mix of deliveries and cannot, of course, deal with disturbances issued by on-process generated occurrences. In order to relax the strict sequentiality, assembly lines with buffers (for temporary storing the work in progress) and with by-passes (for supporting re-routing abilities) could offer intermediate possibilities as compared to the highest flexibility of the job-shop layouts (later discussed, in section 2.5).

As second example development, the case of a plant is considered, which is engaged in the production of families of plastic items, such as dispensers, atomisers, and triggers for home preparations; lips, spouts, and nozzles for ready-drinkable cheer packs and cans; nonrefillable closures for bottles and decanters; and similar nonreusable devices that shall be delivered to buyers, that use such artefacts for distinguishing individual offers. Interested customers require strong characterisation with quick changing features for advertising new products and job orders might be subject to abrupt changes with very short fore-warning. These artefacts have size of about 30 to 50 mm, with four to ten parts to be joined by spring locking (or equivalent operations) and only occasionally, by glue sticking or ultrasonic soldering.

Results of the previous discussion are again recalled to help specifying the individual functional units and defining the proper modules granting each elemental operation. Once the modularity is properly assessed, the assembly facility is worked out combining the proper functional units along a transfer track travelled by the workpieces assorted on pallets. The units are operating at each assembly stand receiving (after singling and orienting) the parts to be joined by suitable feeders. Modularity encompasses physical and logical resources. This time, moreover, special concern is moved on shop logistic to make possible more than one material flow along assembling. The goal is to keep the benefits of flow-shop organisation adding a limited flexibility in order that artefacts with different assembly schedules might be processes without the need of reconfiguring.

The solution is suggested by distributing the application oriented modules at the shop level with a multiple path interconnection. The simplest arrangement replaces the line by a closed loop; workpieces can be diverted to the workstations in any order since they may travel the track more than once. The sophistication could be raised by making use of multiple-loop layouts with on-process supervisors to properly update the dispatching policy. Transfer paths redundancy has, of course, cost limitations; as flexibility approaches job-shop requirements, then the logistic should address different dispatching services (e.g., AGV-based solutions).

Modular Assembling Set-Up with Sorting and Storing Transfer

The case study hereafter considered deals with the production of quite small componentry, i.e., safety closures for spirits bottles, Figure 2.24; the transportation tracks can, thus, be obtained with comparatively cheap outfits. The investigation of the alternative has been undertaken by showing, through simulation, advantages and drawbacks in order to assess, both, technical feasibility and economical return on investments.

The switching to new devices moves through the fitting up of the injection machines with properly shaped moulds and the setting up of the assembly section with suitably planned work cycles. The analysis carried on the functional blocks shows that most of the processing modules (screwdrivers, joining fixtures, clinching rigs, etc.) and of the testing modules (for monitoring, quality control, and acceptation duties) are repeated in the assembly schedules; the main differences exist between the fitting modules (pallets, fixtures, etc.) and the transfer modules (for transportation, feeding, and handling functions). The govern modules are easily endowed by sufficient flexibility to support control and task adaptive programming.

The example case shows that a simple closed-loop assembly line, Figure 2.25, already brings in enough versatility. Modularity is properly exploited for the transfer structures by using already available units, built with standard tracks, completed with forks, by-passes, and spur tracks. Transport is done by belts or chains, with locking-unlocking mechanisms operating on palletised trays. The tracks are easily rearranged, with bend connections, providing the opportunity of modifying the loop or of interlacing more than one loop. The assembly stands are located along the tracks with the local processing modules properly linked and interfaced to the parts feeding devices.

© 2001 by CRC Press LLC

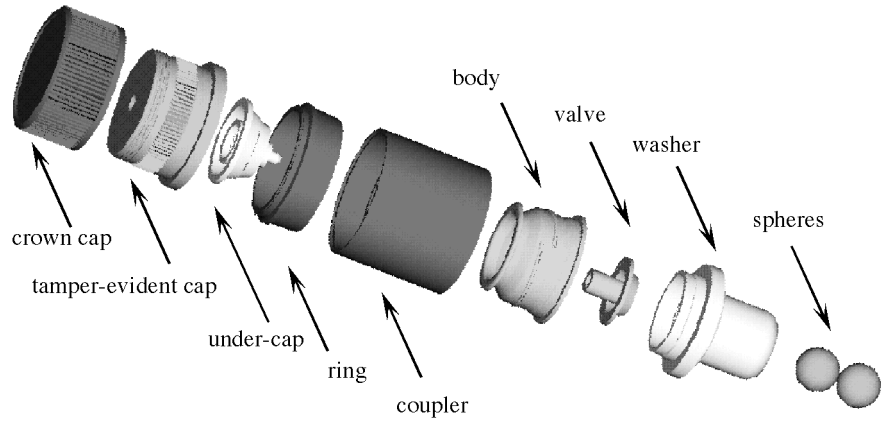


FIGURE 2.24 Exploded view of a typical non-refillable closure for bottles.

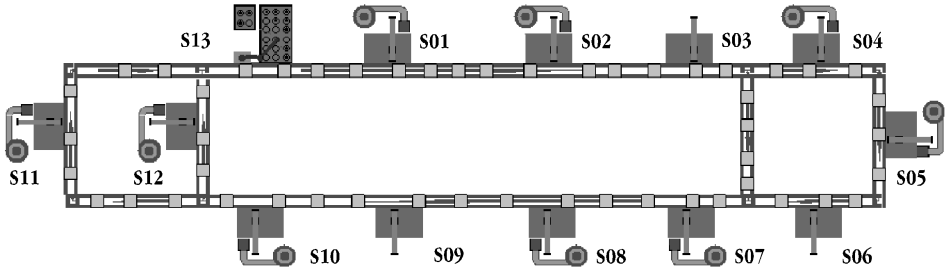


FIGURE 2.25 A closed-loop assembly line with by-passes.

The resulting set-up presents as a fixed automation assembly line, with a single entry and a single output, when an artefact type is assembled alone. It can reach comparable productivity with sequential assembly schedules properly split if the work cycles are balanced at each assembly stand.

The workstations are obtained by combining suitable functional units (and related part-feeders) with pieces positioning and handling rigs, as already described. As before, modularity provides the usual options: addition or removal of modules; inclusion of singling and sequencing devices; change of workstations order; inclusion of monitoring or testing instruments; etc.; moreover, this time, the designer has the possibility of distinguishing the assembly flow (completed by special purpose modules) from the dispatching flow (assured by redundant tracks). The present study assumes that the assembly section simultaneously operates on two families of products: type A item requires 12 operations; type B, only 9 operations. Work cycles, see Figure 2.26, are arranged so that, of the 13 stations included by the section, a subset (S_{01}, S_{02}, S_{03}, S_{07}, S_{08}, S_{09}, S_{10}, and S_{13}) is engaged for both items; a subset (S_{04}, S_{05}, S_{06}, and S_{11}) operates on family A items and, finally, station S_{12} is active for family B items only. The layout of Figure 2.25 shows the by-pass avoiding family B items to go through work stations S_{04}, S_{05}, S_{06}, and the fork to branch off family A across S_{10} and family B across S_{12}.

The branch off could also be used to add spare resources. To such a purpose, the careful analysis should be previously done to acknowledge critical tasks and to arrange beforehand functional redundancy. The investigation requires experimental data obtained from the manufacturing processes. Results are directly related to a product type and bear consistent meaning for sets of artifacts. The proposed safeguard is simply

© 2001 by CRC Press LLC

Assembly Station	Type A closures	Type B closures
S01	Positioning of the ring-coupler group and check of parts presence	
S02	Body spring locking and presence check	
S03	Part overturning	
S04	Positioning of the undercap and presence check	
S05	Pre-screwing of the tamper-evident cap	
S06	Final screwing of the tamper-evident cap and presence/deformation check	
S07	Fixing of the crown cap and presence check	
S08	Part overturning	
S09	Valve positioning and presence check	
S10	Washer fixing and presence check	
S11	Spheres insertion and presence check	
S12		Sphere insertion and presence check
S13	Parts unloading and fixtures' cleanness check	

FIGURE 2.26 Work-cycles of the two families

obtained by duplicating a subset of modules, or, perhaps, by adding a general purpose assembly stand that could be programmed to perform tasks left undone (because of failures or any other reasons). These extra-stations need be reached by the product flow, travelling again the closed-loop track (or running through by-passes or spur tracks). The balancing of the production flows needs buffers to avoid bottlenecks formations.

Simulational Validation of the Out-Fit

The number and location of the extra stations, their outfitting to make possible the management of the recovery flexibility plans, and the preparation of the correct transfer and logistics layout are all topics requiring extended simulation work sessions before assessing advantages and drawbacks of the additional options. This kind of investigation is done by the expert simulator MAA-SIFIP, implemented to give account of the dynamical behaviour of modular adaptive assembly facilities. The code is based on the MODSIM III language of CACI Products Co.; it exploits object-oriented programming and avails itself of a powerful graphic interface: during layout design, Figure 2.27, a friendly window-based model-builder drives the user in the selection of different alternatives and provides on-line congruence checks; defaults apply for most fields and a catalogue of common commercially available equipment can be browsed on-line; a rough costs estimate of the plant is also provided at the end of this stage.

Statistically generated interrupts, giving account of local failures, can be enabled during simulation: a rule-based governor selects the restoring policy, enabling the alternative routing and activating the recovery schedules until the conditions for the original flow shop sequencing are reinstated (with proper refixturing). Runtime plant evolution can be observed by means of an animation restitution; moreover, several pop-up windows allow a complete monitoring of all equipment by simply mouse clicking on the corresponding icon on plant layout. Finally, simulation results are displayed at the end of the simulation run by means of usual charts that show the most important production coefficients, such as throughput, production rate, machinery utilisation, maximum/minimum waiting times, queues lengths, etc.

The reference tool MODSIM III is a high-level programming language that provides both object-oriented features and discrete-event simulation capabilities and is used to build large, process-based simulation models. The language constructs for describing system components have both concurrent

© 2001 by CRC Press LLC

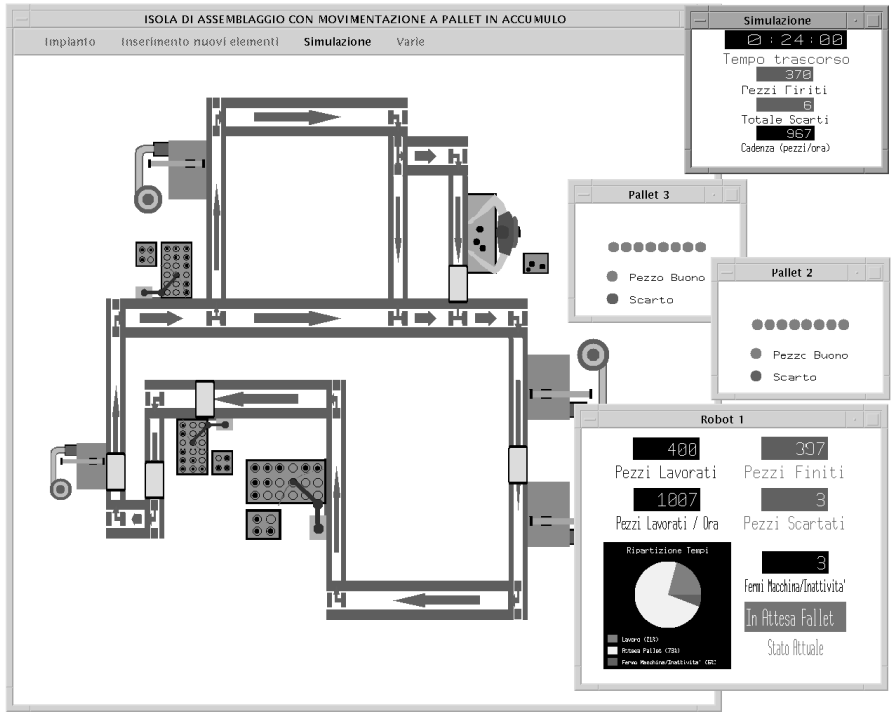


FIGURE 2.27 Runtime animation of the MAA-SIFIP simulation-code.

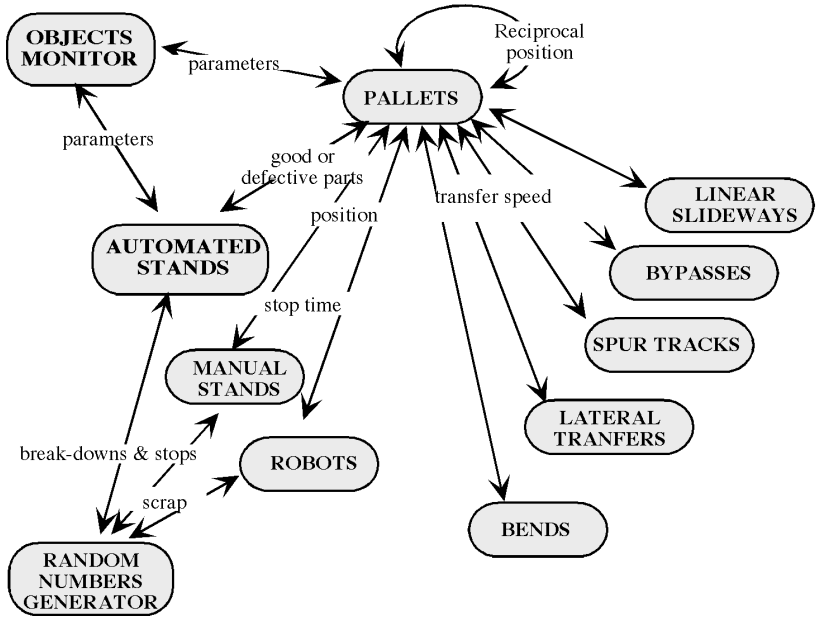


FIGURE 2.28 The organisation of the MAA-SIFIP simulation code.

© 2001 by CRC Press LLC

and interacting behaviours in simulation time. Object coding supports multiple inheritance, encapsulation, and polymorphism. Its complex environment allows the users to develop advanced models that are:

- visual, with extensive use of graphics and animation and capability to easily set up a scenario through icon drag and drop technique;
- interactive, allowing users to monitor the simulation and to interrupt it to change parameters;
- hierarchical, allowing users to select the desired level of simulation detail; and
- portable and open, since it generates readable C++ code and provides an interface to access external libraries.

Programming, leading to the code MAA-SIFIP, deserves, as well, few comments. The specification of function classes exploits object attributes for instances and object methods for contexts, Figure 2.28.

The govern module presents as a logical resource, corresponding to the object describing the control sequences, with individual time ordering attributes given by the assigned tasks the longest interval specifies the span for concurrence and the timing and schedules depend on the selected set-up. The loading and fastening of proper amount of pieces on a pallet is carried over by means of the object corresponding to the local picking and latching device. The stream of the parts to be assembled, after singling and orienting, is ruled (in hidden time) by the feeder object. The process is continuously monitored; the updated figures are provided by means of ordering indices and quality labels. Logic timing is, moreover, referred by a list which is conditioned by procedural knowledge (sets of rules) and a gatekeeper. The governing logic is accessible and the operator can enable alternatives modifying the layout (reset range) or changing the process plan (outfit range).

The preliminary validation of the MAA-SIFIP has been undertaken with the simple set-up obtained by closing a straight assembly line in order to build up a four-sided layout so that unfinished workpieces could be further processed. The arrangement allows pallet accumulation along the transfer chains with buffering figures depending on sides length (or on path travelled between two subsequent assembly stands). The minimal number of pallets is fixed assuming to have two pallets at each stand and an intermediate pallet on the linking path. This is consistent with the four-sides loop with the smallest number of active stands; the facility can be travelled in the shortest time (with minimal work-in-progress), but without buffering capabilities. The failure of one unit immediately affects the incoming flow and a queue trail starts. With larger four-sides loops, travel time increases; queues, however, could establish without affecting upstream stations; the routing of the pieces has to be controlled to feed only consistent workstations.

The present study has taken in consideration the option of a four-sides layout with internal branching tracks, that supplies limited adaptivity coming from modularity, endowed by store-up and by-pass possibilities, Figure 2.25. This avoids the drawbacks of strict task sequencing, still satisfying the leanness checks, since assembly is performed by special purpose devices. The study, within the just stated limitations, has further considered:

- the options provided by the workpieces dispatching flexibility; and
- the extra possibilities offered by the addition of processing units.

The usual schedules are processed reaching productivity figures comparable with special purpose assembly lines; the limited downgrading is due to the increase of the total work-in-progress. Then some unsteady situations are considered:

- first of all, the assembly of artefacts requiring different schedules (number of joining operations and order sequencing) is studied; the closed loop layout makes possible to programme with freedom at which assembly stands the workpieces are processed; productivity, obviously, decreases if the strict sequencing is not followed any more. Comparative simulations are performed with artefacts requiring both shorter and longer assembly cycles; and

© 2001 by CRC Press LLC

- moreover, it is supposed that (alternatively) some unit could be turned into failure mode; if the operation sequencing is not strictly fixed, the workpieces are allowed to bypass the temporarily out-of-order station and complete the assembling at a subsequent loop; the failure mode, normally, depends on the parts feeding mechanisms: thus can be generated at random affecting any of the assembly stands.

Further Issues of Modular Assembly Facility with Redundancy

The code MAR-SIFIP offered an initial overview of the opportunities given by modular assembly cells, supporting easy plant reconfiguration each time case arises to modify the products. The suggestion develops on the capabilities of specialising the assembly sequences and preparing the efficient set of functional modules, each one performing the acknowledged unit operation. By that way, the flexibility becomes a useful opportunity to be fixed off-process. The return on investments simply depends on the capability of quick reconfiguration with appropriate reuse of the different modules. Once the set-up is fixed, the plant reaches fixed automation performance; equivalently, properly parameterised, the code MAR-SIFIP is ready to be exploited as reference issue for the off-process planning of optimal schedules.

The code MAA-SIFIP is supporting a slightly more sophisticated approach for obtaining flexibility during on-process operation. A simple case investigation has considered the joint assembly process of two types of nonrefillable bottle closures, requiring twelve and, respectively, nine unit operations. The simulation results given by the code cover several mixes of the two items and situations can be compared, ranging from either one of the closures alone to more complex mixes, Figure 2.29.

For standard operation conditions with a single product, the code MAA-SIFIP simply authorises workpieces stockpiling. When a module is driven to failure state, the by-pass track is used for the selective dispatching of the pieces. As compared to the flow line solution, the closed path layout makes possible production continuity, with additional investments that are known in advance (and are negligible, when the extra set of functional units are already procured and need only be interfaced to belt). In the practice, open questions are left, concerning configuration and dispatching policy for pieces transfer, possible selection and location of extra resources to add, storing and sorting mechanisms for feeding the right parts to the assembly stands, and the likes.

Answers are not unique. Advantages and drawbacks of possible alternatives can only be assessed depending on the peculiarities of the individual case, that is why simulation is very important to show the conditions for the returns on each addition. The example validation tests just recalled already give a first insight on a very simple arrangement, where the redundancy is based on the transport service, by exploiting a shunting side-way to separate the assembly path of the two items and a paralleling by-pass to distinguish the functional tasks. The recovery capability is entrusted to the closure of the assembly path; then, when some assembly operations are missing, the restoring is obtained by adding the possibility of recycling the pallets bearing uncompleted workpieces. Results already show that the flexibility on the dispatching system is worth to avoid situations that lead to critical stops with open loop assembly lines.

The basic features of modularity are well understood if consequences are duly assessed. The availability of expert simulation is a powerful decision support providing hints towards innovation. By means of off-process simulation, setting (and resetting) of the facility is studied, whenever artefact changes are required; the process effectiveness is assessed during the stage of product development. The on-process management of flexibility needs the on-line versatility of, both, the processing units and of the transfer dispatching. Indeed, if a defective unit stops the regular completion of the scheduled sequences the production of an assembly line breaks off. To prevent the effects of resources failure, job shop arrangement and robotised assembly stands should be available, supplied, and fixtured by addressable transfer services and fed with the correct kit of parts to be joined. Such extended flexibility (considered in the following section of the chapter) is paid by lower productivity; therefore, large spreads of solutions should be studied (by simulation) before fixing the attention on any option.

© 2001 by CRC Press LLC

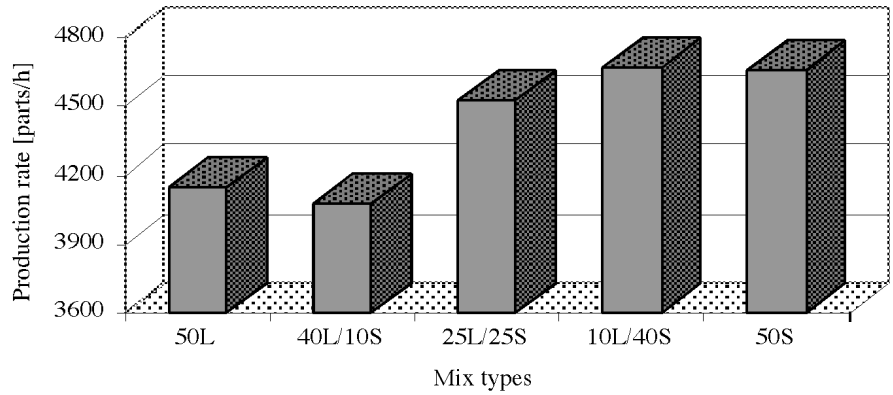


FIGURE 2.29 Production rate for different mixes of 50 pallets (S: short cycle, L: long cycle).

Simulation tests can, moreover, consider the opportunity to duplicate single (critically loaded) assembly stands; these extra resources could be located on a parallel side-way (as compared to current work cycles) and travelled only at unexpected occurrences. Finally, virtual reality investigation can deal with limited changes of the assembly schedules, in order to make possible to process new artefacts (provided that the requested functional units are present) pointing out extra issues, such as

- the assembly of artefacts having smaller number of components is performed, with small productivity decrease, covering the loop with a single run along shunted paths; and
- the assembly of artefacts, requiring to simultaneously join two items with similar duty cycles, can be done by travelling the loop with run selection at the dedicated forks.

2.5 High-Versatility Self-Fixturing Facilities

Economy of scope issues mean allotment of resources within the primary process, flexible specialisation, decentralised manufacturing, shortest time-to-market, quick return on investments, customer driven satisfaction, trade liableness due the delivery certified quality, and the likes. These present as conflicting aspects, unless the scope is balanced to the particular application, in terms of lean engineering check-ups. Checks presume full visibility of enterprise's functions, with concern of flexibility effects. Through intelligent manufacturing, new improvements are expected from the set-up of resources (CFC frame) tailored to product mixes included by the enterprise strategic planning and the fit-out of schedules (MDM frame) adapted to the production agendas along the enterprise tactical planning. By concurrent emulation-simulation, actual running conditions of flexible facilities can be tracked; the productivity figures are, therefore, evaluated for each strategic, tactical, or execution time span.

Market trend now leads to shorter and shorter items life; this makes investments into special purpose resources questionable requiring amortisation plans to be prosecuted with exactly the same product. Efficiency by on-line adaptivity and on-process flexibility is the innovation brought forth by the economy of scope, contrasting earlier trends to oligopoly formation according to the rules of the economy of scale. Hints have been introduced in sections 2.2 and 2.3, with emphasis on general criteria to specify flexibility effects. It is understood, indeed, that information technology and robotic equipment provide tools to extend on-process flexibility ranges; the accomplishments of intelligent manufacturing, however, still lag behind the forecast achievements, as they were extrapolated only some years ago. It has been inferred that technology anticipate needs, but only proper balancing of investment against efficiency, deferred to lean engineering rules, leads to return.

© 2001 by CRC Press LLC

For such a reason, to focus attention on cost effectiveness, the discussion is restrained at shop floor level, contrasting efficiency and versatility in terms of the ability of dealing with extended and quick-changing product mixes to deliver time-varying orders within the expected due dates. Even at the lowest level of flexible automation, relevant innovation needs be secured for production operators' commitment:

- assembly robots need, usually, satisfy technical specifications more sophisticated than efficiency and versatility alone; accuracy and dexterity become relevant properties (these aspects are, however, not explicitly included by the present discussion); and
- computer aids enabling joint control and management of flexible automation require: **(**a**)** to preserve the causal (nonanticipatory) dependence among structured entities to simulate the transformations of the material processes; **(**b**)** to exploit heuristic inference abilities, to emulate anticipatory responses of the judgmental decision logics.

Techniques in computer-integrated assembly become basic option for cost effectiveness when versatility is inherent property of facilities and the higher exploitation of resources is deferred to the on-line and on-process adaptation of production plans aiming at just-in-time delivering of buyers-driven artefacts. The techniques signify the centrality of expert simulation, this time, not only as consultation aid for choosing the assembling set-up (the CFC setting, as discussed in the previous section), but as well as governing aid for enabling return by the on-line ruling of strategic, tactical, and execution flexibility (by means of MDM fitting). Software capabilities remain, of course, instrumental contribution only; for the reference cases of the section, hardware abilities, thus, have a presentation akin with the one developed for the previous section examples.

Modularity was assessed as implement to reach versatility by plant resetting. Robotics is recognised as technology providing functional versatility during plant activity. For developments aiming at the modularisation of the assembly facilities, the analysis into subtasks is performed to singularise individual functional bent according to properly defined actions. A similar analysis is required for the design of the functional attributes that robotic equipment needs be endowed, including, this time, the integrated hardware and software constraints. Information technology further has to provide govern-for-flexibility requirements for adaptive assembly schedules. The functional analysis of the work cycles, summing up, is accomplished to obtain the reference data to address the development toward: versatility by re-setting the programming and control functions or toward: versatility by re-fitting the planning and adaptivity abilities.

The first solution is useful when:

- (comparatively) small batches of (quality) products have to be assembled according to changing users' driven requirements; and
- facility refixturing and programming are positive options and unattended work shifts may prosecute with (simple) supervisors forcing the facility behaviour to preserve given preset patterns.

The second solution is required when:

- complexity of artefact mix to be concurrently joined is high and modularisation would require too many special purpose functional units; and
- separation in time of the different series of products is not feasible and flexibility has to be exploited on-process by adapting on-line the behavioural patterns depending on the emerging occurrences.

Two developments are commented in the following, considering the case leading to the CFC setting with account of simultaneous engineering paradigms joined to the versatility of the robotic equipment and, separately, the case aiming at the MDM fitting by integrated control and management of intelligent facilities through an expert assembly governor. Again, as in the previous section, the set-up of the facility versatility if first referred to the choice of the resources that, usefully, have to be included to reach the production capacity consistent with the enterprise's programmes. With the second example, the study moves on to the options given by the on-process management of flexibility for recovery purposes in front

© 2001 by CRC Press LLC

of emergencies. The concept of emergency is widened to cover several situations including, in particular, the programmed strategy of switching between varying production agendas, to deal with the mix of artefacts that are purposely assembled by the same fixture.

Robot-Operated Assembling Set-Ups

As first case development, that profits of an integrated approach, the robotic assembly of buyer's driven artefacts is discussed. The presentation refers to sets of comparatively expensive special pieces to be contributed as instrumental outfits of customised products; these parts can be forwarded for further processing by other sections of the enterprise or supplied to external end-manufacturers for the build-up of equipments on a one-of-kind base depending on the needs of the final users. The fabrication of the intermediate components faces the typical issues of flexible specialisation: products mix should be modified according to the integrators' technical requests; items quality needs be steadily granted and outrun the competitors standards; and due time and delivery might require sudden acceleration of the production scheduling to win new orders. Batch size and artefact quality figures make unsuitable the modularisation into units, that could be used again on condition of plant modifications; robotics, therefore, presents as worthy option provided that investment in technologies satisfies the leanness checks.

Resort to robotised assembly sections grant enterprise's effectiveness, when versatility covers the expected production variability and autonomy makes possible unattended work-shifts at peak requests. To reduce the time-to-market by just-in-time schedules, the CFC frame has to be redesigned while doing the conceptual development of the devices to be assembled. The investigation, thus, yields to a sample validation of simultaneous engineering, by exploiting design-for-assembly procedures and by relating plant performance figures to the quality-function deployment policy.

For the present discussion, let us refer to the family of multiple-connector devices shown in Figure2.30 and consider production plans for supplying sets of items ranged by type and differing according to the requirements of the integrators that use them as special parts of customers' adapted equipments. The assembly facility, when organised for the concurrent design of product-and-process, is placed in the position of behaving as active supplier capable of cooperatively designing new devices as case arises and manufacturing them with transparent quality figures for the subsequent integration into final artefacts.

The Assessment of the Automated Assembly Cell

The robotic assembly of these simple, but prized, components is assessed with the help of the special purpose simulator DAX-SIFIP, developed using, as standard reference, the package ROANS by JHF. This is a PC-based modular software devoted to describe automated work cells; it offers options for: 3D animation; kinematic and dynamic analysis; off-line programming of industrial robots and engineer's support for new robots design; and visual restitution capability to create virtual reality experiments (on the simulated process, on the testing equipment, etc.). When interfaced through the proper hardware it may enable simple control actions according to user needs.

The principal properties of these kinds of packages need consider: the software has to be used off-line to assess the layouts and schedules suitability; or transferred as front end tool for robot programming and calibration; span: the code should permit to rapidly evaluate competing options, reducing errors and allowing wider range of choices; leanness: simulation needs significantly reduce costs by finding out problems since the planning stage, instead of discovering them during the setting stage; and focusing: details should be investigated to select modules or connecting frames, to optimise fixtures behaviour, to compress the duty cycles, etc.

Basically, the simulator DAX-SIFIP helps to perform the following actions:

- to build-up the automated cell geometry: the user can create any object by a 3D editor, or import it from standard CAD developments (dxf and iges file formats); then, he can compose the cell by listing the objects in it; finally, he can verify if the robot reaches every point in the cell space, without interfering with other objects;

© 2001 by CRC Press LLC

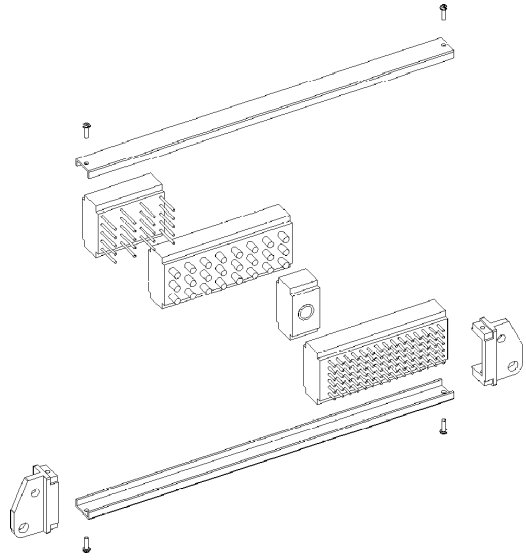


FIGURE 2.30 Exploded view of the special purpose multiple-connector assembly.

- to test processes by virtual reality simulation: the user can work in an interactive way with the graphic representation of the cell; kinematics routines calculate the cycle time and verify for interferences and collisions. The user can verify the correctness of his multitasking programming logic by simulating inputs and PLC communications;
- to analyse data: all the information concerning position, velocity, and acceleration are presented by the kinematic module; inertias, torques, internal forces, RMS power, duty cycle, and other motor parameters are computed by means of the dynamic and drive analysis module;
- to communicate between robots: several post-processor modules are available aiming at translating ROANS routines into robot languages and vice-versa; ROANS has procedures to calibrate both cell and robot. There is also the possibility to monitor robots by reading their joint positions from the serial port; and
- to perform direct programming: a tool has been created in which DAX-SIFIP operates as programming unit and trajectory generator for a given axis controller board. The user may graphically create a new virtual robot and the tool will act as controller, with the valuable opportunity of merging reality and virtuality, through on-line and off-line programming supported by the direct access to the controller memory.

By means of the ROANS based DAX-SIFIP simulator, the reference set up of the assembly cell happened to be composed by two SCARA robots, five auxiliary fixtures, and a test unit that measures product's properties while assembled. Work and test cycles are performed by a hierarchic supervisor, combining monitoring, assessment, and command functions. The supervisor is interfaced to a graphical display, supplying on-progress situations, with emergency warnings whenever they arise. The cell can remain in the operation mode for a full unattended work shift (8 hours), when an order completion is scheduled. This makes the assembly facility on-duty over the 24 hours span once the cell is initialised and production has not to be modified.

Let refer to a typical assembly cycle. The assembly robots select the items from the feeding station (properly supplied by an AGV). The final devices are assembled, using pallets for carrying the individual workpiece, by putting together over than ten parts. For the case application, the steps of the operation sequence may be characterised as shown in Figure 2.31.

© 2001 by CRC Press LLC

Operation	Robot ID
1. U-shaped base positioning	1
2. Left end-plate positioning	2
3. Gripper change	1
4. Screw-1 fitting	1
5. Right end-plate positioning	2
6. Screw-2 fitting	1
7. Sub-assembly reorientation	2
8. Connector-blocks placing	2
9. U-shaped crosspiece positioning	2
10. Screw-3 fitting	1
11. Screw-4 fitting	1

FIGURE 2.31 Schedule of the multiple-connector assembling.

The Setting of Robot Programming and Control Functions

The setting of the assembly agenda exploits data flow programming. The execution order is determined by the flow between blocks; multiple paths or action concurrence are thus easily established. To ensure specific execution order, for and while loops, or case statements are defined for sequential, repetitive, or branching connections; the options are mentioned on the displayed icons. Through the modular hierarchy, details and complexity are hidden at lower layers, while retaining the access to the information during the initial setting, for programming, or, at emergency, for performing recovery operations. The processing ability of generating PC-based virtual equipment is valuable at each cell resetting, since it eliminates low-level programming for interconnecting the functional blocks and for selecting the appropriate command operativeness.

The analysis of the flexibility requirements leads to the appropriate setting, by means of an out-of-processcharacterisation, indeed the sequencing figure is considerably high for the robotic assembly; and the complexity figure could vary depending on the final artefact, by requiring, sometimes, combined stokes to perform elemental tasks. The operation timing presents sequencing with partial task overlapping. The supervisor enables the assembly during unattended work shifts, by synchronising the robot with the auxiliary fixtures for plugging in, fitting up, and screw-driven connections. The overall architecture of the assembling cell is shown by Figure 2.32.

While doing the cell setting, the redesign of the artefacts should be profitably studied, in order to improve the duty cycle for automatic assembly (the simultaneous insertion of block sequences, for instance, is a trick to drastically simplify automatic assembly, but it requires a new architecture of the frame). The cell fitting is, thereafter, concerned by integrating command and monitoring. The control needs assure job sequencing and coordination by means of decentralised PLC (acting on the pallets supplier, the screwing fixtures, the pins drivers, the friction press, etc.) connected to the robot governing module. This module is chosen to have several programming options (from keyboard or from display through icons and mouse); the duty cycle is managed by the cell supervisor with monitoring and steering functions.

The final set-up corresponds to a fully instrumented cell with sensors on the vibratory feeders and on robot grippers. Feeders are monitored through optical fibre transducers that recognise component type and orientation; the process stops in case of unavailability of the correct part. The check of the component on the gripper is performed by an optical gate; the robot repeats the fetching task at a warning message. The testing equipment operates within a computer integrated environment, that performs signal acqui-

© 2001 by CRC Press LLC

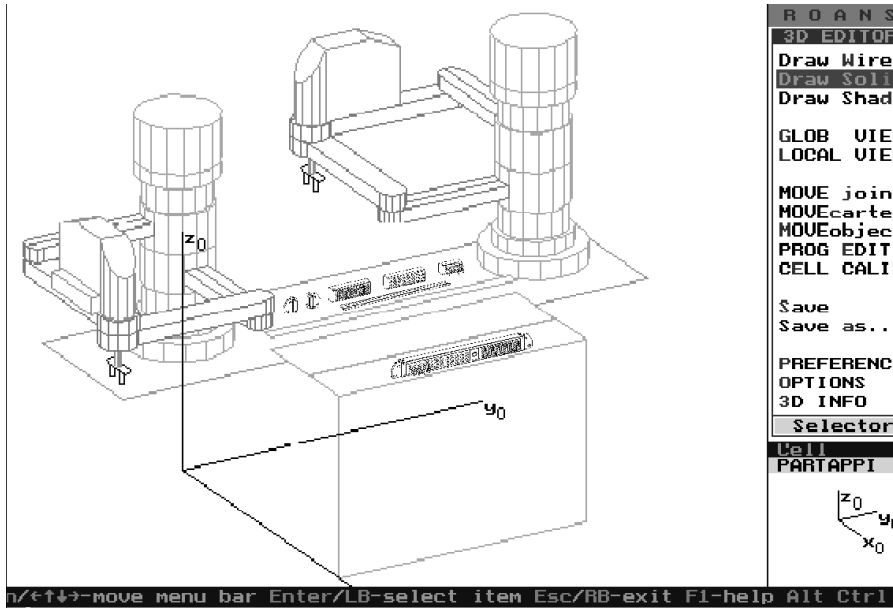


FIGURE 2.32 Model building environment with layout of the robot-operated assembly cell.

sition and processing by means of virtual equipment, properly specified for speeding up both the programming tasks and the diagnostical data acknowledgement.

The user interface is based on standard software with link to external application blocks; the implementation of such requirements of the assembly cell is also studied by means of the simulation code DAX-SIFIP, established on the detailed functional model of the overall (assembling and testing) processes. The simulation aids, supplying effective graphic restitution, give hints on the concurrent activity of the two assembly robots and become a development tool assuring the effective virtual reality environment for robot programming and for supervisor setting.

The Supervisory Process Monitoring

Robotic assembly for the present case application is useful to expand versatility at the strategic range and to enable supervision at the execution range. The assembly cell operates under a hierarchic govern; the resulting layout is reset at the strategic and at the execution ranges. This kind of solution permits to increase the assembly effectiveness of the considered individual items, obtained also by the automatic assembling during the unattended work shifts. The diversification is deferred to the integrated development of product and process, done by previous experience and skill of the connectors designers, driven and coordinated by the requests of the (subsequent) parts integrators. The versatility of the robotic cell is the basic prerequisite of running for new orders; then the delivery of repetitive artefacts is left to the supervisor's controlled assembling along with unattended work shifts.

The cell supervisor rules the assembly and testing operations and assures the ordered execution of the planned agendas, according to bottom up requests. The user interface avails itself of suitably arranged charts, allowing quick recognition of new products and simple fitting for modified technical specifications. A proper block is developed for data acquisition, conditioning, and restitution with built-in ability of providing virtual equipment generation; of performing on-line analysis and control functions; and of supplying interactive front panels for virtual reality setting operations. The program, Figure 2.33, exploits the software modularity, to create virtual equipment and subequipment and to refer to icons for the

© 2001 by CRC Press LLC

```
PROGRAM  1
...
CTYPE CARTES
TOOL 0,0,40,0,0,0
[position of TCP by respect to the
TAP]
SPEED J,100
[speed set-up]
ATTACH BASE
[relative position of BASE]
MOVE J,C4R
[moveto BASE]
GRASP TOOL1
[TOOL1 grasps the BASE]
MOVE J,C10I
[move to destination place]
RELEASE TOOL1 BASE
[TOOL1 releases the BASE]
START I1=1
[Wait for robot 2]
...
```

```
PROGRAM  2
...
CTYPE CARTES
TOOL 0,0,-120,50,0,0
[position of TCP by respect to the
TAP]
SPEED J,100
[speed set-up]
ATTACH END-PLATE
[relative position of END-PLATE]
MOVE J,C7R
[moveto END-PLATE]
GRASP TOOL2
[TOOL2 grasps the END-PLATE]
MOVE J,C9I
[move to destination place]
RELEASE TOOL2 END-PLATE
[TOOL2 releases the END-PLATE]
...
```

FIGURE 2.33 Pieces of the programming code for the cell supervisor (synch. commands).

build-up of the monitoring and the supervising jobs; functional blocks of the outer layer are accessed, at the front panel to provide interactive actions; to specify the acquisition patterns; to select the analysis routines; to enable the network and file I/O operations; to address the command sequences; to monitor the current diagnosis data; etc. The inner layers include blocks for data conditioning, curve fitting, signatures detection, trends recording, statistical analysis, controls setting, occurrences linking, and the likes.

Computer-integration of the assembling facility aims at real time coordination; as data are acknowledged, the display directly provides monitoring and supervising functions. Tests are initialised by robots as soon as the assembled artefact is available at the stand and are aimed at verifying both functionality and performance. Measurements are reported into a file, corresponding to a step of the approval cycle; results are displayed (pins layout of the elemental blocks, electrical insulation between tested circuits, etc.) and provide a visual assessment of the required functions.

The preparation of assembly sequences, the scheduling of work-cycles and testing agendas, the management of diagnostical frames, etc. are established more efficiently by means of standard languages; thereafter, these external blocks need only be linked into the data flow diagram, as specialised application of the multitasking system. The overall procedural knowledge for this third case investigation is therefore coded according to application-driven requirements and fixed into procedures based on the off-process assessment of optimal assembly plans. The unattended work shifts are performed within the frames of fixed automation; the supervisor simply monitors processes progression to provide review data on the products and hints for possible betterments.

Assembling by Integrated Control-and-Management

The last case investigation deals with a situation requiring on-line flexibility. Efficiency is, now, principally sought by means of MDM fitting (and not mainly by CFC setting, as before) aiming at the integrated control and management of the available facilities through an expert governor, directly interfaced to the assembly facility. Production plans are concerned by extended mixes of continuously modified items. The study considers an integrated section having multipurpose automated assembling stations that work

© 2001 by CRC Press LLC

in parallel, each one operated by a handling robot. Effectiveness is obtained by adapting the schedules to ongoing situations and by integrating tactical and execution ranges. This results to reach:

- high productivity by decentralised optimal control, during the steady (tactical horizons) spans; and
- recovery ability at the (strategic or unexpected) discontinuities, by means of supervisory management.

The investigation has to be carried on by means a knowledge-based software for the functional design of assembly cells layout and, thereafter, for the integration of real-time management and control options. After the initial facility set-up stage, the subset of the software blocks performing the decisional tasks is used for production planning, aiming at govern-for-flexibility issues, while assuring the monitoring support for the continuous agendas fitting and facility expansion. Resources behavioural properties are, this time, critical requirement. The individual station delivers variable mix of finished artefacts, assembling different sets of parts. Moreover, since product types and assortments evolve according to users' requests, several artefacts need be concurrently assembled on condition of supplying each work station by properly adapted time-varying dispatching. For the case discussion, let consider three families of final artefacts (plastic throttle valves), Figure 2.34, each one presenting intermediate assemblies.

The valves have symmetric end connectors (already fitted up with related ball seats) and union nuts. The assembly cycle, Figure 2.35, assumes to position the body, to insert a connector, to screw a union nut, to turn over the body, to insert and properly position the inner ball, to insert the second connector, to fasten the second nut, and to fix the handle. The cycles do not considerably differ depending on valve sizes and series; thus the same work stand can be used to assemble different valves even if like products are better processed on the same posture to simplify outfit trimming and parts feeding logistics.

The balancing of the manufacturing agendas leads to an assembly section with three stands, Figure 2.36, each one operated by a robot and provided with automatic parts feeders supplied by a common shuttle operated transporter. Each robot is devoted to the mounting of a specific valve series, unless the special setting of the priority indices modify the ongoing attribution. The supervisor should check the priority setting and the due date requirements for the delivering of the assorted batch. The parts feeding is preset through the shop floor dispatching service. The robots perform either the picking of components or their insertion and locking.

The assembling section used in this final example case, Figure 2.36, is operated by three robots: each one is devoted to the mounting of a specific valve. The valve-1 has priority over the others: in case of failure of the robot-1 the production of the valve-1 is moved to the section-2 (thus stopping valve-2 processing). The robots perform either the picking of the subcomponents (moving to the proper buffer pallets) or their mounting. Shuttle transporters, driven by shop floor logistic, are used to unload the empty pallets and to replace them with new ones.

The Computer Integrated Aids

The considered flexible facility, aiming at assembling a time-variable mix of items, has been investigated by means of the expert simulator XAS-SIFIP based on the shell G2 by GENSYM. The tactical scheduling faces the simultaneous assembling, in the example case hereafter summarised, of several families of throttle valves, differently combined to grant due time delivering of the assorted batches, as required. The valves are obtained by joining PVC components and subassemblies and their sample assembly cycles of three valves are collected in the table of Figure 2.35.

The complexity of the situations faced for the integrated control-and-management of the example assembly process requires a comparatively sophisticated programming aid. The knowledge-based tool G2 was therefore used, since it possesses very friendly access and it is further capable of supporting real-time operations. This means that the expert governor enabled for the simulation, when directly interfaced to the physical facility, performs the control-and-management actions of the selected policy.

G2 is a development environment for creating and deploying intelligent real-time applications and is, today, extensively used for factory automation: since it is well known, in the following only details related to the particular application are recalled. It provides a complete, graphical development environment for

© 2001 by CRC Press LLC

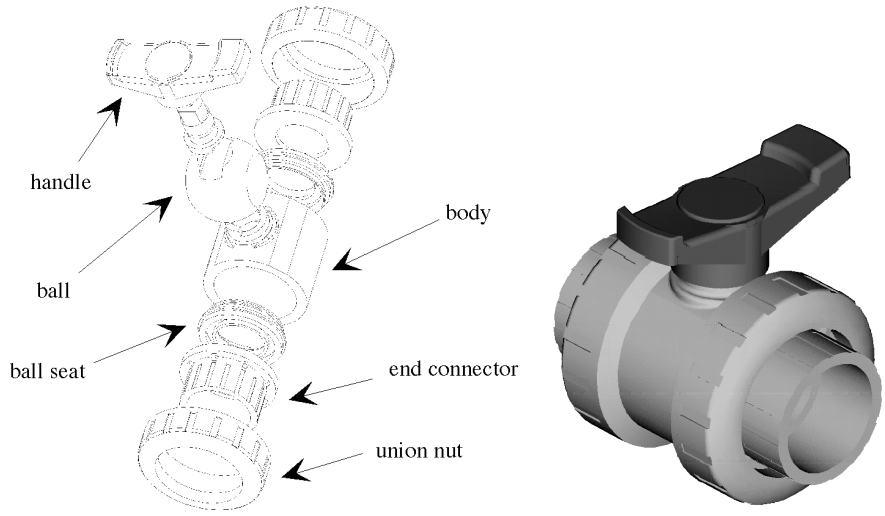


FIGURE 2.34 3-D views of reference samples of PVC throttle-valves.

Operation	Time (s) V1	Time (s) V2	Time (s) V3
1. Body positioning	30	25	25
2. Right end connector positioning	10	7	5
3. Right union nut screwing	20	15	15
4. Ball insertion	20	10	8
5. Handle fixing	15	10	7
6. Left end connector positioning	10	7	5
7. Left union nut screwing	20	15	15

FIGURE 2.35 Assembly cycle for the example PVC valves.

modeling, designing, building, and deploying knowledge-based applications, with a friendly editor performing on-line syntax checks; the developments are based on object-oriented design and knowledge representation is mantained and extended through classes in the G2 hierarchy. The dynamics of concurrent processes can easily be coordinated by using an occurrence driven timing, so that the tool is well suited to real-time applications, such as: monitoring, diagnosis, and automatic handling; supervision and control; intelligent network management; etc.

The package XAS-SIFIP purposely makes use of object-oriented programming, to describe, Figure 2.37, both physical and logic resources; this grants easy modification and facilitates future expansion. The govern module includes rules and procedures for encoding heuristic or, respectively, causal conditioning knowledge. Occurrences are transient object classes and the related instances are created for situational acknowledgements and destroyed once processed; important attributes are the current time, that assures sequencing and delivering description; and the identifying label that specifies the actor type and the resource attribution. The simulation is also based on relational links to allow indirect object addressing. The availability of specialised conditioning knowledge opens useful opportunities.

- Rules are introduced, typically, for activating sets of tasks which create inquiries with assigned values and connections to objects and enable procedures each time situations arise for the instanciation of actions dependent on the acknowledged occurrence. A different group of rules performs feasi-

© 2001 by CRC Press LLC

© 2001 by CRC Press LLC

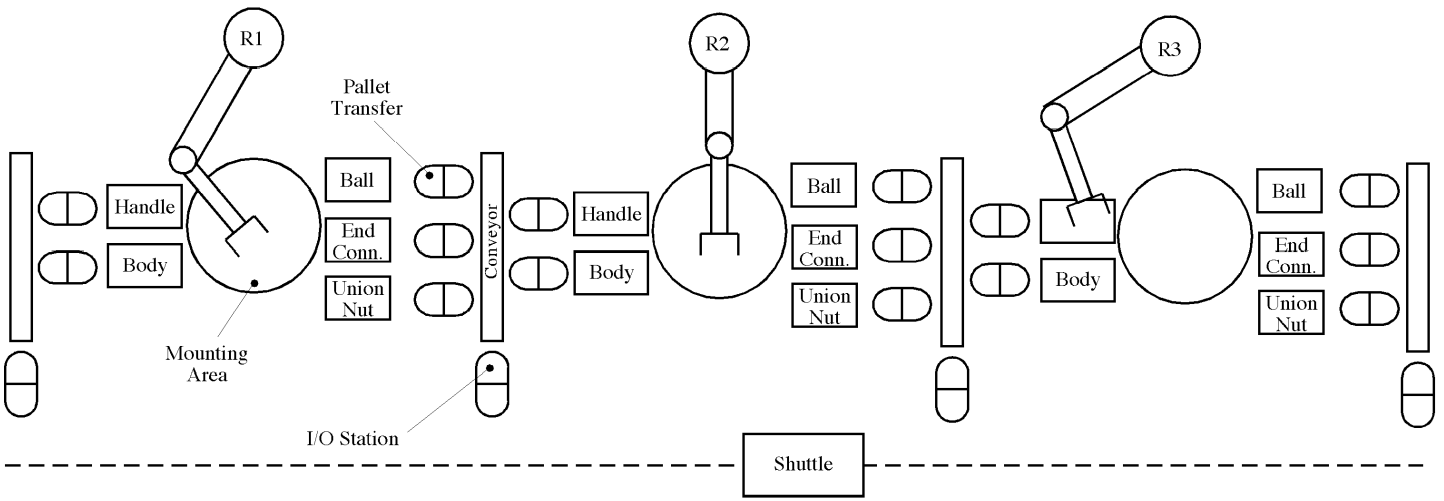


FIGURE 2.36 Reference layout of the flexible assembly section.

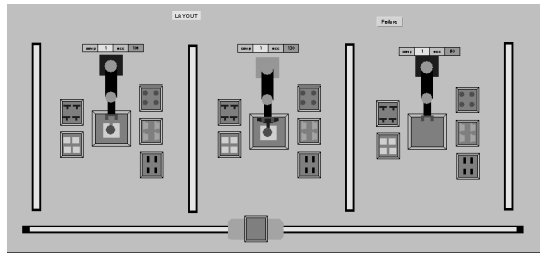


FIGURE 2.37 Visual display of the XAS-SIFIP simulator.

bility tests on requests, depending on the current plant situation; checks make use of predefined service procedures, having in charge the instanciation of the objects to be addressed by individual request. Several rule patterns are considered: initial rules, to activate the knowledge base; whenever rules, for rule connection; if rules, for conditioning concatenation; etc.

- Procedures are used for the execution of search and reorder algorithms of lists each time the set of actions to be fired is logically assessed (e.g., data sorting, display modification, simulation time updating, etc.). Specialised procedures are appended to modify the requests or to assess the request feasibility.

The knowledge of the package XAS-SIFIP has been dislocated into several workspaces by grouping objects according to types and functions. Appropriate subworkspaces gather specialised objects to organise the commands or to assemble variables and rules. The knowledge partition, by dynamically activating workspaces and subworkspaces, assures modularity, for modifying or extending the structural CFC and the behavioural MDM frames. A pertinent workspace is defined for visual restitution, Figure 2.38. The assembly section layout is represented by icons, for the I/O units, workstations, work-in-progress, etc. The simulation outputs are further provided by several readout-tables distributed, in relation to each physical resource, into the pertinent subworkspace.

The Development of the Integrated Assembling Facility

The XAS-SIFIP code, being based on the high level shell G2, is properly suited for real-time developments. The flexibility effects can be monitored and measured with continuity. Attention is focused on the effects induced by:

- robot characteristics: tasks are analysed and overall duty times are acknowledged as averaged figures, since more details can be omitted once the teaching is completed;
- failure patterns: the recovery strategies are investigated for different breakdown or damage situations and modified workstation priorities; and
- compositions of mixes to be assembled: the (three) artefacts have different assembly times, thus effectiveness depends on the assortments of the delivery agendas.

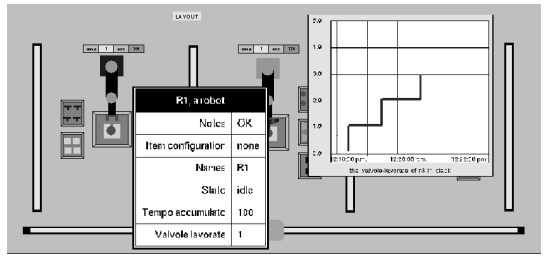


FIGURE 2.38 Typical output work-spaces for data display.

© 2001 by CRC Press LLC

To develop the assembly facility, the resources behavioural properties have to be assessed for all product types and assortments included over the strategic horizon of the enterprise programmes. For any particular product type and assortment, as a first instance, the association station-and-family is done. Parts, subassemblies, and products are supplied with properly assorted feeding. The handling robots withdraw the components to accomplish station loading/unloading. The assembly section presents as a set of concurrent processes, with connection constraints and rhythm marked by the robot services. The handling operations (pick, transfer, place) are governed by heuristic rules to update the scheduled delivery, according to the current situational contexts, maximising the workstations operation time (or minimising the work-in-progress). Efficiency fitting is concerned by the behavioural MDM frames to enable govern-for-flexibility issues.

The assembly operations are given in charge to an expert governor. The development of such a module requires the previous validation of the returns on investments. This is assessed through simulation, performed by means of the XAS-SIFIP code, which allows the measurement of the process productivity figures, for assigned strategic choices; the confrontation of govern-for-flexibility options, for selecting optimal MDM fittings; and the experimentation on the governing module, incorporated on-line for factory automation (unattended mode).

Initially, the study is concerned by the development of the facility, aiming at the selection of both physical and logic resources. The code XAS-SIFIP is used as consultation aid to support the comparison among strategic choices, tactical alternatives, or recovery options. The governing policy avails itself of priority criteria readily modified depending on delivering schedules requests. Productivity is maximised setting the components feeding strategies to reach highest operativity of the assembly stations. During the simulation, failure and repair occurrences are statistically generated, with driving occurrence series extrapolated from previous experience; the switching to the recovery strategies is performed after checks on the work-in-progress distribution and on the sought delivering agendas. Similar checks are done, when the unit is ready to resume the operation state. Thereafter, just-in-time plans are tracked while preserving due date requirements.

Time evolution, during simulation, is occurrence driven. The updating is done by a scheduler that acknowledges new occurrences and selects the prosecution according to judgmental rules, Figure 2.39. Occurrences are placed in a list; the state is continuously updated in relation with the acknowledged situation. The simulation cycle starts with the occurrence at the top of the list, performs the requested tasks that might generate new occurrences to be inserted in the list, withdraws the completed occurrence, and gets ready for a new cycle. The management of the strategic, tactical, and recovery flexibility requires judgmental abilities performed by an expert governor based on an inference motor for heuristics reasoning in order to possess emulation functions duplicating the decisional logic. The resulting governor has knowledge-based architecture, with the suitably coded rules and procedures selected by a gate-keeper, depending on the ongoing context.

This gate-keeper, or inference motor, is based on several options. Rules are evoked with given scan interval, via forward or backward chaining as fixed by the appended attribute; in function of the assigned focal object or focal class; in relation with chosen rule pattern; each time the occurrence mentioned by the antecedent verifies; and whenever a reference variable reaches the prearranged threshold. The internal branching enables, by default, the actions evoked by each rule; at the end of the rule, or at any given time, suitable in order procedures need be specified, to suspend the activation. A procedure recalled by a rule, moreover, originates the tasks considered by the inference motor at the end of the rule; it follows that the supervisor could be requested to face sets of different tasks; thus, ordering requirements are necessary to grant the evolution consistency.

The On-Process Integrated Controller-Manager

The govern fitting is performed through extensive exploitation of expert simulation, by means of the XAS-SIFIP code; this includes the subsets of blocks for on-process and on-line operations, later transferred, as factory automation software, to supervise to the real plant operations. Once a design-feedback-

© 2001 by CRC Press LLC

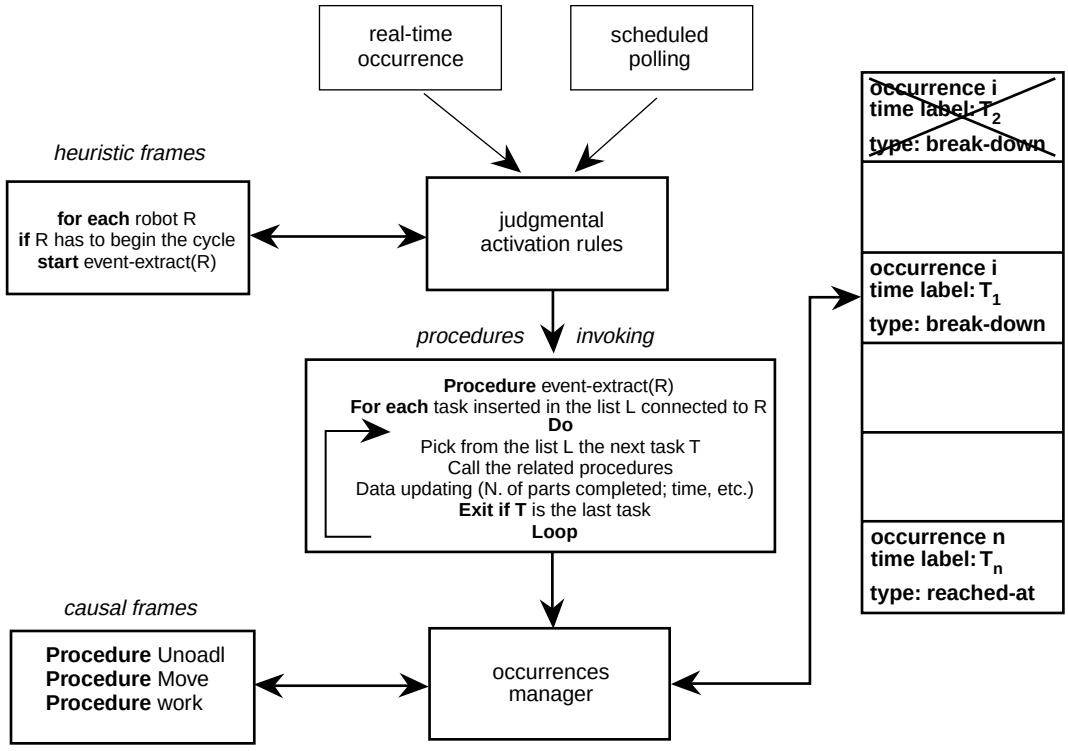


FIGURE 2.39 Organisation of the decisional scheme.

choice cycle is completed the governor is ready to operate on-line. The basic requirement, distinguishing the on-process exploitation of flexibility from the previously recalled set-up phase, is the need of real time operativity. With the knowledge-based architectures of the expert simulator XAS-SIFIP, developed by means of the shell G2, a twofold advantage arises:

- the implementation of heuristic blocks is straightforward and the emulation of human-like decisional platforms provides the software instruments for enabling the real-time factory automation with govern-for-flexibility features; and
- the goal: flexible automation is coherently obtained by merging patterns of action and patterns of structure, provided that structures carry the signature of the originating changes and actions grant the consistency of the generated configurations.

Flexible automation is built by equivalent lore bases. On the knowledge-based actions-patterns side, one looks after expert governors which transform control actions according to current process data, on condition that the process structure consistency is preserved; the resulting issues lead to the integrated controllers and managers, exploiting real-time decisional manifolds, directly operating on the assembly processes. On the knowledge-based structures-patterns side, reconfigurable build-ups, combining the setting of the CFC structural frames with the fitting of the MDM behavioural frames, are faced; the related issues provide knowledge support, Figure 2.40, to close the decision cycles and to reset higher performance assembly layouts.

Adaptive governors present multiple-layer build-ups with vertical connections among relational, generative and information layers, interacting with the process, see Figure 2.17. The relational layer exploits patterns of structures and the expert governor expands on preprocessing modules for fixing the CFC and

© 2001 by CRC Press LLC

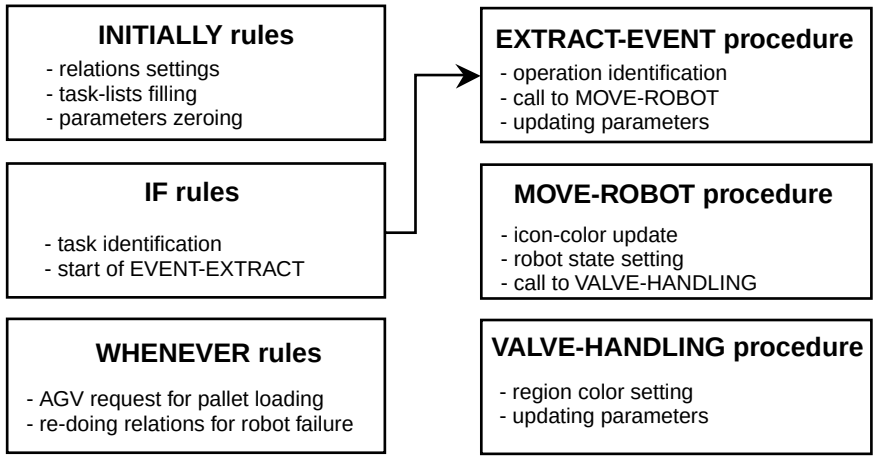


FIGURE 2.40 Organization of the procedural knowledge-based architecture.

the MDM frames, with interfaces for structural model setting and for behavioural mode fitting. The generative layer exploits patterns of actions and propagates the causal responses with algorithmic blocks while acknowledging consistent suppositions through heuristic modules. The decision supports achieve best exploitation of flexibility by incorporating on-process data from the information layer; this monitors the performance evaluation figures and provides the diagnosis of facts for enabling trend-monitoring (proactive) maintenance. The closure of the decisional cycles for control and management of assembly operations makes extensive use of heuristics, and effectiveness of a policy is assessed by measuring the actually reached performance.

Intelligent manufacturing leads to the distinctive innovation of integrating control-and-management at shop floor operations level. The return on investment is not fully predictable, since the capability of processing varying (and, perhaps, still not yet designed) mixes of artefacts is the characterising feature of the innovation. Effectiveness, indeed, does not depend only on the technological versatility of robotics; it mainly depends on the ability of making profit from balancing strategic, tactical, and executional flexibility. The issue is, practically, achieved by referring to expert programming tools, such as the code XAS-SIFIP, assuring consistent view for both off-process virtual reality assessments and on-process govern-for-flexibility implements.

Effectiveness through Flexibility and Adaptivity

The trends toward worldwide manufacturers, established on concurrent enterprise setting and fitting, are accepted as challenge for future competitiveness. The evolution is mainly based on knowledge-intensive organisations and effectiveness is looked for by enabling the economy of scope, instead of the economy of scale. Production engineers will be required, therefore, to face the completely new problems of reaching return on investment through govern-for-flexibility issues, that maximise productivity by means of diversified artefact mixes, over the strategic horizons; optimal schedules, on the tactical horizons; and recovery actions after emergencies, at the executional horizons. These new approaches require the full visibility of every relevant process variable; the models of the assembling facilities need be pushed to every useful detail. Some further aspects of the simulation aids are, therefore, considered.

The integrated assembly section is travelled by assorted components used to rationalise the transfer tasks. Every workstation delivers finished artefacts belonging to any of the scheduled families; for efficiency, the specialisation is planned, but not strictly obliged: the handling robot can freely perform feeding and retrieval operations. The coding aid bears similar adaptivity. With object-oriented programming, each resource is specified by identifiers (selected icons); structural attributes (topological location, cell

© 2001 by CRC Press LLC

geometry, etc.); functional characteristics (robot performance, buffer bed space, assembly times, etc.); and operation state (free, working, out-of-order, busy, idle, etc.), with colour-code allegation and logic updating to state the situational progression. For instance, the artefacts to be assembled are defined not-existing, when components are still separated; waiting, when completed within the workstation; in transport, during the robot handling; and available, when ready on the delivering pallet.

The expert simulator XAS-SIFIP is provided by sets of post-processing modules for data storing, statistical indices reckoning, performance evaluation and, in general, for helping the cross-coupled assessment of the issues obtained by each consistent CFC choice or by any possible MDM frame. By properly exploiting the statistical modules, combined with the monitoring of the selected performance indices (net production, activity ratio, etc.), decision aids can be generated, to suggest how facilities should be reset, how assembling cycles might be rescheduled, whether different fixtures should be included, whether jobs could be simplified by a modified valve design, etc. The simulation, for instance, helped to modify the robot feeding service in order to speed up the pick-and-place jobs, reducing the overall cycle-time.

The validation of control and management policies requires the emulation of the decisional logic, jointly with the simulation of the material processes. With the XAS-SIFIP code, the first is based on the expert shell G2, the second is written in the C language; the communication in-between is based on the linking interfaces, Figure 2.41. The concurrence of the cooperating parallel processes is situation-driven and the governing rules are modified as case arises. The experimentation has provided useful hints for settling competition between facilities and balancing parallel flows and for rationalising the vertical sequencing and reducing the work-in-progress. The possibility of conflicts among the different agents acting during the assembly process has been considered in the design of the simulator and is properly modelled. As a matter of fact, programmed or random interactions of each cell with surroundings can require the rescheduling of the production or the enabling of real-time recovery actions. In view of these features, the granulation of the simulator has been chosen as thin as suited to deal with this kind of events and object-oriented programming has been extensively used to manage the concurrent time evolution of the different agents competing for the limited resources of the cell; the decisions, on the other hand, have been kept in charge of the distributed cooperating expert nuclei to foster parallelism.

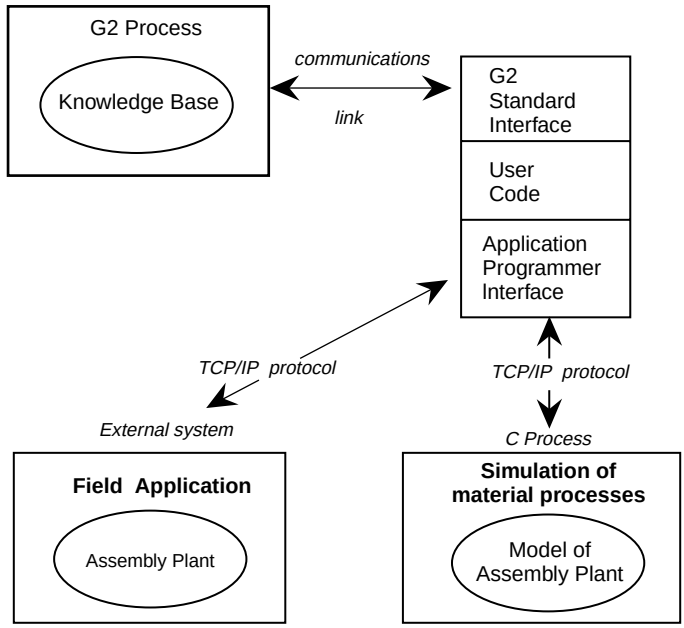


FIGURE 2.41 Organization of the real-time control-and-management.

© 2001 by CRC Press LLC

2.6 Concluding Comments

Assembly tasks of customer-oriented artefacts are important aspects, often representing more than one third of production costs. The benefits of intelligent manufacturing are already recognised; assembly sections, in particular, profit of making use of knowledge intensive settings to support factory automation. In the chapter, the ability of dealing with the pertinent knowledge is contrasted with the enterprise's goals for the off-process setting of the facility effectiveness (CFC frames) and, respectively, for the on-process fitting of the facility governing logic (MDM frames).

Current investigations stress on the relevance of planning and scheduling and on their integration with artefacts design and with processes govern in view, in particular: of the decreases of time-to-market, of lot sizes, etc.; of the improvement of products life time, of the recycle and dismiss options, etc.; aiming at providing flexibility. Assembly facilities can be faced at different stages: design, planning, scheduling, management, control, up-keep, withdraw, etc.; flexible automation provides critical support at shop floor level, when the other stages are already fixed with consistent figures. Return on investments, indeed, has to be assessed quantitatively; this is done by the choice of benchmarks for the resetting (CFC frames) and refitting (MDM frames) of the assembly processes, in order to reach:

- completeness: benchmarking should be concerned by the technological features of the overall products mix that, potentially, might be assembled;
- consistency: the standardisation should refer to actual industrial contexts; ideal set-ups need be avoided when correspondences with reality are missing; and
- simpleness: the figures shall have straightforward interpretation; that is why functional models and computer simulation are basic references.

According to the developed ideas, effectiveness plays an important role when related to actual industrial situations. Assembly facilities should be application driven and the joint development of products-and-processes according to design-for-assembly rules shall be performed on a recurrent decision cycle base. Computer-simulation is, then, a standard aid for fully enabling the advantages by testing alternatives before the shot off of new layouts. The updated assessing of plant layouts is explicitly dealt with by two (of the four) discussed cases, studied by the MAR-SIFIP and, respectively, the DAX-SIFIP codes.

The first situation considers a manufacturer of cheap devices that need be delivered by huge batches and could abruptly be modified according to users' wishes. Profitability is obtained by mass production, joined to the possibility of switching to new artefacts with very short time-to-market. The solution is given by modularity in the assembly facility, making feasible very quick reconfiguration of fixed-automation (high-productivity) layouts. The situation described by the DAX-SIFIP code belongs to rather different industrial issues; it aims at on-line assessing and sequencing the tasks for small batches artefacts to be integrated into one-of-a-kind final products. The high versatility of robotised cells is exploited for quite straightforward production programmes. Computer integration is, here, confined to the ability of performing monitoring and steering functions, that need quick and easy resets at each new batch, but could be propagated unaltered, particularly along the unattended workshifts, for completion of the already planned batch.

Drawbacks of off-line planning in front of unexpected occurrences are inherited by both example developments. The inclusion of on-line adaptivity shall be considered (at the level of the MDM frames) for the on-process management of flexibility. Modularity, indeed, could suggest not only the opportunity of versatility management by means of off-process plant reconfiguration, rather as well the opportunities of gradually including on-process flexibility. A limited set of options is, first, investigated with the MAA-SIFIP code; the study, moving from the modularity of the processing and the transferring units, aims at exploring the versatility of a conveniently arranged dispatching service to make acceptable the assembly of (slightly) varying product mixes.

Most manufacturers are simultaneously concerned by diversified production, as option to be used to satisfy clients' needs, by combining standard items (perhaps, suitably reorganised), minimising additions and modifications. The complexity of the assembling sequences makes computer simulation a cheap

© 2001 by CRC Press LLC

short cut for obtaining useful data for checking the flexibility effects on the productivity. The XAS-SIFIP code has been developed for such (fully general) situations; it can be used as:

- consultation aid, for quality and efficiency upgrading of products and production; it provides the comparative assessment of actual performances of flexibility, over different (strategic, tactical, and operational) horizons, for consistent delivery agendas;
- governing option, for enabling multiple actions: assuring efficient parallelism of the concurrent assembly processes, during the steady tactical horizons; and supervising to the decision manifold, during the transient (initialisation or restoring) periods.

The techniques in computer-integrated assembly have been investigated in the chapter at the level of shop floor automation. The cost effectiveness of solutions heavily depends on the ability of preserving leanness, i.e., of monitoring the return on investments by a value chain of artefact's quality transfer, so that material addition and functional option are verified to really increase enterprise's competitiveness. The criteria lead to the permanent exploitation of expert simulation, to select the facility set-up (CFC frames) suitably matching the enterprise's strategies, when productivity is the main constraint for competitiveness. Visibility of knowledge-intensive assembly schedules and transparency of flexibility effects (MDM frames) are options brought forth by intelligent manufacturing, when extension and variability of product mixes are primarily concern to expand the enterprise's rentability, as balanced issue, by combining strategic, tactical, and execution ranges. These different features are shortly reviewed as concluding comments of the chapter by showing the extra options of flexibility and the related issues of the coupled effects on productivity.

Off-Process Setting of a Customer-Driven Mass Production Assembly Facility

The first example shows how modularity could grant return on investments and short time to market, when effectiveness requires very high productivity aiming at delivering cheap artefacts, properly following customers' specifications. The facility versatility is set and managed off-process exploiting the modularity of the processing resources. The assembly process is, previously, fully analysed into its elemental functions and proper modules are developed. The choice of the effective solution is obtained using the package MAR-SIFIP. Expert-simulation is, for the case application, pertinent reference to iterate the decision cycles: "design/setting, testing/feedback, choice/fitting," aiming at optimising products and processes each time new artefacts are defined to win extra orders.

The production is possibly subject to abrupt changes and fixed automation efficiency conflicts with delivering modified items with continuity. On-process adaptive management of current assembly schedules is unfeasible by simple modularisation; of course, the options can be included by replacing individual (special purpose) modules by the appropriate versatility robotic equipments. The resulting solution obtains flexibility by functional redundancy and programming capability.

Exploiting Recovery Flexibility with Adaptive Modular Assembly Facilities

Sometimes the on-process flexibility is suitably attained preserving the functional resort of modular units and looking for alternatives by the adaptive management of the shop-logistic within an assembly facility based on (modular) resources redundancy. The option is discussed by the second sample case, with comments on merits and worthies of modularity applied to logistics adaptivity, as compared to more sophisticated means for providing flexibility.

The second example develops the idea of pieces dispatching adaptivity, with the addition of (inexpensive) functional redundancies to make possible the exploitation of the recovery flexibility, even with a plant granting the productivity of the special purpose automation. Centrality is given by shop-floor logistic flexibility. The addition of store up and by pass capabilities requires sophisticated planning of material transport flows. With computer simulation, supported by the package MAA-SIFIP, several alternative solutions are easily compared. Further to the dispatching flexibility, the study has also considered the cheap solution of duplicating the critical processing units; more sophisticated solutions call for the inclusion of robotised assembly stands that might temporarily accept different workpieces.

© 2001 by CRC Press LLC

Programming Optimal Assembly for One-of-a-Kind Products

The third example suggests the relevance of process monitoring and how automation could be exploited to reduce costs and increase quality. The market share would widen, once reached this combined goal, requiring capability of further diversifying family types as new buyers are won. The set-up of a convenient assembly cell needs, however, higher complexity technological resources, not easily reduced to modular units. Then robotics is winning option, with beforehand check of versatility for future potentialities, to preserve leanness with return on investment. The objective is assessed with proper computer aids for setting the CFC frames adapted to the assembling of families of oriented devices that can readily be incorporated into customised high quality final artefacts.

The DAX-SIFIP code helps selecting the assembly cell, with standard equipment (robots, feeding units, cooperative rigs, testing instruments, etc.) suitably supervised by a programmable environment supporting virtual equipment generation. It has already proved the capability of reacting to market diversified requests. The set-ups are selected to automatically face individual orders with optimal assembly schedules. Unattended work shifts are commonly enabled with suitably prearranged fabrication agendas that (at the moment) do not require refixturing operations during the shift. The switching to new orders presumes a check on the CFC frames with the off-process consultation aid of simulation; then the highest productivity is transferred on-process and enabled all over the scheduled tactical horizons.

Efficiency Through Tactical and Recovery Flexibility

The last example discusses how to deal with integrated control and management strategies for plant effectiveness. The simulation code XAS-SIFIP, in this case, is not only a consultation reference, but also provides the decisional support to the govern-for-flexibility block. Assembly section performances are measured by the continuous monitoring of work progression. Flow-shop schedules are enabled on the tactical horizons, minimising the waiting intervals at workstations and balancing the parallel processes. As a cell switches to a damaged situation, the assembling of the related item is temporarily stopped and transferred, by the emergency procedures, to other stations (better served by the robot), until the recovery strategy enables an alternative scheduling. The steering logic remains in any case transparent, so that the effects of flexibility are fully measured.

The discussion has introduced to adaptive planning and scheduling solutions based on the performance evaluation of flexible assembly facility obtained by expert simulation. The modelling encompasses products and processes and requires the joint assessment of structural and behavioural frames. Specialised decision supports are needed also for on-process managing the plant flexibility. The resulting computer-integration ranges at a comparative high level of sophistication; investment needs be joined to lean engineering check-ups, to justify the returns, provided by integrated control and management, to reach effectiveness by the exploitation of recovery flexibility at emergencies; tactical flexibility to enable optimal schedules during steady horizons; and strategic flexibility for planning customer-driven production agendas.

2.7 Acknowledgments

We gratefully acknowledge the financial support of CNR (Italian Research Council) for the basic developments of these studies leading to the implementation of series of SIFIP packages under the frame of the projects PFTM (Progetto Finalizzato Tecnologie Meccaniche) and PFR (Progetto Finalizzato Robotica). We also thank all the manufacturing companies that cooperated with us for several developments, including the example applications of sections 2.4 and 2.5, by providing all relevant information: Connei S.p.A. (Sestri Ponente, Genova), Controlli S.p.A. (Sant'Olcese, Genova), FIP (Casella, Genova), Montech SA (Torino), etc.; we also mention the companies Ravasi Robotics S.r.L. of Brivio (Lecco) and San Giorgio System Technology (Sestri Ponente, Genova) for their courtesy and for having specially supported the use of the tools ROANS and TAYLOR, along with the present investigation.

© 2001 by CRC Press LLC

References

[ACA86] Acaccia, G.M. et al., 1986, X-SIFIP: a knowledge-based special purpose simulator for developing flexible manufacturing cells, *IEEE Intl. Conf. Robotics & Automation,* pp. 645–653, San Francisco.

[ACA87a] Acaccia, G.M. et al., 1987a, Rule-based expert-govern for flexible manufacturing, *IEEE Intl. Conf. Robotics & Automation,* pp. 558–567, Raleigh.

[ACA87b] Acaccia, G.M. et al., 1987b, Knowledge-based simulators as programming facilities in production engineering, *2nd Intl. Conf. Computer-Aided Production Engineering,* pp. 71–78, Edinburgh.

[ACA87c] Acaccia, G.M. et al., 1987c, Expert-simulation: a tool for exploiting flexible manufacturing, *ASME Conf. Computers in Engineering,* vol. 2, pp. 227–235, New York.

[ACA87d] Acaccia, G.M. et al., 1987d, Knowledge-based simulators in manufacturing engineering, D. Sriram and R.A. Adey, Eds., *Computational Mechanics Pub*, Unwin Brothers, pp. 327–344, Southampton.

[ACA88a] Acaccia, G.M. et al., 1988a, Failure analysis and maintenance model for a flexible manufacturing cell, *7th IASTED Intl. Conf. Modelling & Control,* pp. 80–86, Grindelwald.

[ACA88b] Acaccia, G.M. et al., 1988b, Information data-based structures for flexible manufacturing simulators, *Advances in Production Management Systems,* A. Kusiak, Ed., Elsevier, pp. 649–662, Amsterdam.

[ACA88c] Acaccia, G.M. et al., 1988c, Knowledge-based programming instruments for robotized manufacturing facilities, *Intl. J. Advanced Manufacturing Technology,* vol. 3, n. 3, pp. 53–66.

[ACA89a] Acaccia, G.M. et al., 1989a, Expert-simulation of a tool-dispatcher for factory automation, *Intl. J. Computer Integrated Manufacturing,* vol. 2, n. 3, pp. 131–139.

[ACA89b] Acaccia, G.M. et al., 1989b, A distributed-interconnected control for computer integrated manufacturing, *Intl. J. Computer-Integrated Manufacturing Systems,* vol. 2, n. 2, pp. 108–114.

[ACA89c] Acaccia, G.M. et al., 1989c, Development of decentralized controls in flexible manufacturing, *Intl. J. Computer Applications in Technology,* vol. 2, n. 2, pp. 89–100.

[ACA89d] Acaccia, G.M. et al., 1989d, Shopfloor logistics for flexible manufacturing based on distributed intelligence, *Intl. J. Advanced Manufacturing Technology,* vol. 4, n. 3, pp. 231–242.

[ACA89e] Acaccia, G.M. et al., 1989e, An expert-scheduler for the tool-stock management in a CIM-environment, *Intl. J. Advanced Manufacturing Engineering,* vol. 1, n. 4, pp. 203–210.

[ACA89f] Acaccia, G.M. et al., 1989f, Design of intelligent governors for material-handling equipment of automated factories, in *Software for Factory Automation,* T. Sata and G. Olling, Eds., pp. 297–312, North-Holland, Amsterdam.

[ACA92a] Acaccia, G.M. et al., 1992a, Govern-for-flexibility of tool-stock and delivery operations for factory automation, in *Information Technology for Advanced Manufacturing Systems,* G.J. Olling and Z. Deng, Eds., Elsevier, pp. 483–490.

[ACA92b] Acaccia, G.M. et al., 1992b, Computer-intelligence in flexible manufacturing, in *Industrial Applications of Artificial Intelligence,* J.L. Alty and L.I. Mikulich, Eds., pp. 344–349, North-Holland, Amsterdam.

[ACA93] Acaccia, G.M. et al., 1993, The govern-for-flexibility of manufacturing facilities: an explanatory example, *Intl. J. Computer-Integrated Manufacturing Systems,* vol. 6, n. 3, pp. 149–160.

[ACA95] Acaccia, G.M. et al., 1995, Pilot CIM plant for lean engineering experimentation, *Intl. J. Computer-Integrated Manufacturing Systems,* vol. 8, n. 3, pp. 185–192.

[ACA96a] Acaccia, G.M. et al., 1996a, Simulational assessment of a modular assembly facility, *3rd Intl. Conf. Concurrent Engineering & Design Automation,* pp. 37–42, London.

[ACA96b] Acaccia, G.M. et al., 1996b, Development of modular assembly facilities with store-up and by-pass management, *8th Europ. Simulation Symposium,* pp. 289–293, Genova.

[Ada88] Adams, M., 1988, Automated kitting for assembly, developments in assembly automation, *Conf. Proc., IFS*, pp. 499–513.

[AlJ93] Alting, L. and Jorgensen, J., 1993, The lifecycle concept as a basis for sustainable industrial production, *CIRP Annals,* 42/1, pp. 163–167.

© 2001 by CRC Press LLC

[AlL95] Alting, L. and Legarth, J.B., 1995, Lifecycle engineering and design, *CIRP Annals,* 44/2, pp. 569–580.
[AND83] Andreasen, M.M. et al., 1983, *Design for Assembly,* Springer Verlag, London.
[Ara93] Arai, T., 1993, Future assembly system in automobile industry: human friendly line, *Intl. Conf. on Assembly,* pp. 9–17, Adelaide.
[ArG85] Arnstroem, A. and P. Grondahl, 1985, A system for automatic assembly of different products, *CIRP Annals,* 34/1, pp. 17–19.
[ArG88] Arnstroem, A. and Grondahl, P., 1988, Advantage of sub-batch principle in flexible automatic assembly as used in the IVK-KTH concept mark II, *CIRP Annals,* 37/1, pp. 9–12.
[ARN90] Arnstroem, A. et al., 1990, Mark II for air motor assembly: an industrial case study, *11th Intl. Conf. Assembly Automation,* MS90-825, pp. 1–15, Dearborn.
[AZU88] Azuma, K. et al., 1988, Cell production system for assembly, *Proc. Manufacturing International,* vol. 2, pp. 101–106, Atlanta.
[ArG93] Arnstroem, A. and Grondahl, P., 1993, Mark III: a hybrid flexible automatic assembly system for small batches, *Intl. Conf. on Assembly,* pp. 137–146, Adelaide.
[BAL91] Baldwin, D.F. et al., 1991, An integrated computer aid for generating and evaluating assembly sequences for mechanical products, *IEEE Trans. Robotics Autom.,* vol. 7, n. 1, pp. 78–94.
[Beh86] Behhuniak, J.A., 1986, Flexible automated assembly: a competitive weapon, *7th Intl. Conf. Assembly Automation,* pp. 367–374, Zurich.
[Bjo89] Bjorke, O., 1989, *Computer-Aided Tolerancing,* ASME Press, New York.
[BoA92] Boothroyd, G. and Alting, L., 1992, Design for assembly and disassembly, *CIRP Annals,* 41/2, pp. 625–636.
[BoD84] Boothroyd, G. and Dewhurt, P., 1984, *Design for Assembly,* Penton/IPC Inc., New York.
[BoK88] Boerma, J.R. and Kals, H.J.J., 1988, FIXES, a system for automatic selection of set-ups and design of fixture, *CIRP Annals,* 37/1, pp. 433–446.
[Bon90] Boneschanscher, N., 1990, On task assegnment in a multi-robot flexible assembly cell, *11th Intl. Conf. Assembly Automation,* MS90-811-1-12, Dearborn.
[BOO82] Boothroyd, G. et al., 1982, *Automatic Assembly,* Marcel Dekker, New York.
[Boo87] Boothroyd, G., 1987, Design for assembly and design for manufacture, *CIRP Annals,* 36/2 (Round Table), pp. 435–437.
[BoR68] Boothroyd, G. and Redford, A.H., 1968, *Mechanized Assembly,* McGraw-Hill, London.
[ChG88] Chase, K.W. and Greenwood, W.H., 1988, Design issues in mechanical tolerance analysis, *Manufacturing Rev.*, vol. 1, n. 1, pp. 223–232.
[ChW94] Chakrabarty, S. and Wolter, J., 1994, A hierarchical approach to assembly planning, *IEEE Conf. Robotics and Automation,* vol. I, pp. 258–263.
[Cuc89] Cucchi, M., 1989, Robotics in integrated systems for vehicle assembly, *20th Intl. Symp. Industrial Robots,* pp. 387–395, Tokyo.
[DEF89] DeFazio, T.L. et al., 1989, Aids for the design or choice of assembly sequences, *IEEE Conf. Systems, Man and Cybernetics*, Cambridge.
[Des93] DeSwaan, A.H., 1993, Object orientation in simulation, *European Simulation Symposium,* pp. 694–699.
[DeW88] DeFazio, T.L. and Whitney, D.E., 1987–88, Simplified generation of all-mechanical assembly sequences, *IEEE Trans. Robotics Autom.,* vol. 3, n. 6, pp. 640–658, and vol. 4, n. 6, pp. 705–708.
[Din93] Dini, G., 1993, A module for the automated planning of grasps in robotised assembly operations, *CIRP Annals,* 42/1, pp. 1–4.
[DiS92] Dini, G. and Santochi, M., 1992, Automated sequencing and subassembly detection in assembly planning, *CIRP Annals,* 41/1, pp. 1–4.
[Dre82] Drexel, P., 1982, Modular flexible assembly system "FMS" from Bosch, *3rd Intl. Conf. Assembly Automation,* pp. 129–153, Boeblingen.
[Eba94] Ebach, H., 1994, Advanced dismantling systems: an important step in lifecycle, *2nd CIRP Conf. on Life Cycle Engineering,* pp. 267–274, Erlangen.

© 2001 by CRC Press LLC

[ElR87] Eloranta, E. and Raisanen, J., 1987, Evaluation and design of plant layout by simulation, *Intl. Conf. Simulation in Manufacturing,* pp. 11–21, Torino.
[ERI93] Erikson, T. et al., 1993, Creating the preconditions for a stepwise automation of the assembly process: a case study, *24th Intl. Symp. Industrial Robots,* pp. 851–857, Tokyo.
[ErO94] Ericsson, A., and Onori, M., 1994, An application with the MARK III assembly system, *25th Intl. Symp. on Ind. Robots,* pp. 25–27, Hannover.
[Eve93] Eversheim, W., 1993, Cost model for assembly, *26th Intern. Symp. on Automotive Tech. and Automation (ISATA),* Aachen.
[Fuj88] Fujita, N., 1988, Assembly of blocks by autonomous assembly robot with intelligence 'ARI,' *CIRP Annals,* 37/1, pp. 33–36.
[Fuj90] Fujimori, T., 1990, Development of flexible assembly system "SMART," *21st Intl. Symp. on Ind. Robots,* pp. 75–82, Copenhagen.
[GIU91] Giusti, F. et al., 1991, Robotized assembly of modular fixtures, *CIRP Annals,* 40/1, pp. 17–20.
[GIU94] Giusti, F. et al., 1994, A reconfigurable assembly cell for mechanical products, *CIRP Annals,* 43/1.
[Gor90] Gordon, S.J., 1990, On-line reconfigurable parts feeder, *ASME 2nd Conf. Flexible Assembly Systems,* pp. 127–131, Chicago.
[GUI93] Guinee, J.B. et al., 1993, Quantitative lifecycle assessment of products, *J. Cleaner Prod.,* vol. 1, n. 1, pp. 3–13, and vol. 1, n. 2, pp. 81–91.
[Gus84] Gustavson, R.E., 1984, Computer-aided synthesis of least-costs assembly systems, *14th Intl. Symp. Ind. Robots,* Gothemburg.
[HEN90] Henrioud, J.M. et al., 1990, Evaluation and selection of assembly plans, *APMS '90, Espoo,* pp. 212–219.
[HIR83] Hirabayashi, H. et al., 1983, Travelling assembly center, *13th Intl. Symp. Industrial Robots,* pp. 20.19–20.31, Chicago.
[Hir88] Hirakawa, H., 1988, Advanced parts orientation systems has wide application, *Assembly Automation,* IFS, 8–3, pp. 147–150.
[Hoe89] Hoekstra, R.L., 1989, Design for automated assembly: an axiomatic and analytical method, *SME Tech. Paper,* Detroit.
[HoL91] Homem de Mello, L.S. and Lee, S., Eds., 1991, *Computer-Aided Mechanical Assembly Planning,* Kluwer, Boston.
[HoS89] Homem de Mello, L.S. and Sanderson, A.C., 1989, A correct and complete algorithm for the generation of mechanical assembly sequences, IE*EE Conf. Robotics and Automation,* pp. 56–61.
[JIA88] Jiang, W. et al., 1988, Computer-aided group fixture design, *CIRP Annals,* 37/1, pp. 145–148.
[JOV93] Jovane, F. et al., 1993, A key issue in product lifecycle: disassembly, *CIRP Annals,* 42/2, pp. 651–658.
[KAN93] Kanai, S. et al., 1993, The integration of computer-aided assembly sequence planning and mechanical design of the assembling machines, *14th Intl. Conf. Assembly Automation,* pp. 95–107, Adelaide.
[Kir92] Kirch, W., 1992, New robotic applications in transmission assembly, *23rd Intl. Symp. Industrial Robots,* pp. 185–190, Barcelona.
[Koj90] Kojima, Y., 1990, CAD application for assembly system and jig, *11th Intl. Conf. Assembly Automation,* MS90-815-1-14, Dearbon.
[KOJ94] Kojima, F. et al., 1994, Manufacturing system design simulator based on facility lifecycle model, *Advan. Intelligent Production,* pp. 81–86.
[KoK96] Kopacek, P. and Kronreif, G., 1996, ROBPLAN: semiautomatic planning of robotised assembly cell, *27th Intl. Symp. Industrial Robots,* pp. 385–390, Milan.
[KoL87] Ko, H. and Lee, Y.T., 1987, Automatic assembly procedure generation from mating conditions, *Computer-Aided Design,* vol. 19, n. 1, pp. 3–10.
[LaE92] Laperrière, L. and El Maraghy, H.A., 1992, Planning of products assembly and disassembly, *CIRP Annals,* 41/1, pp. 5–9.

© 2001 by CRC Press LLC

[LeA85] Lee, K. and Andrews, G., 1985, Inference of the position of components in an assembly (Part II), *Computer-Aided Design,* vol. 17, n. 1, pp. 20–24.

[Lee89] Lee, S., 1989, Disassembly planning by subassembly extraction, *3rd ORSA/TIMS Conf. Flexible Manufacturing Systems,* pp. 383–388.

[LeG85] Lee, K. and Gossard, D.C., 1985, A hierarchical data structure for representing assemblies (Part I), *Computer-Aided Design,* vol. 17, n. 1, pp. 15–19.

[LeS90] Lee, S. and Shin, Y.G., 1990, Assembly planning based on subassembly extraction, *IEEE Conf. Robotics and Automation,* pp. 1606–1611, Cincinnati.

[LIB88] Libardi, E.C. et al., 1988, Computer environments for the design of mechanical assemblies: a research review, *Engineering with Computers,* vol. 3, n. 3, pp. 121–136.

[LiH93] Little, D. and Hemmings, A., 1993, Scheduling automated assembly system, *Assembly Automation, IFS,* 13-4, pp. 8–13.

[Lil66] Lilley, S., 1966, *Men, Machines and History,* International Publishers, New York.

[Lot86] Lotter, B., 1986, Planning and implementation of flexible assembly cells, *7th Intl. Conf. Assembly Automation,* pp. 1–9, Zurich.

[Lot89] Lotter, B., 1989, *Manufacturing Assembly Handbook,* English Ed., Butterworths, London.

[MaF80] Makino, H. and Furuya, N., 1980, Selective compliance assembly robot arm (SCARA), *1st Intl. Conf. Assembly Automation,* pp. 77–86, Brighton.

[MaF82] Makino, H. and Furuya, N., 1982, SCARA robot and its family, *3rd Intl. Conf. Assembly Automation,* pp. 433–444, Boeblingen.

[Mak74] Makino, H., 1974, An experimental approach to NC assembly center, *Proc. Intern. Conf. on Production Engineering,* pp. 486–491, Tokyo.

[Mak89] Makino, H., 1989, The comparison between robotic assembly line and robotic assembly cell, *10th Intl. Conf. Assembly Automation,* pp. 1–10, Kanazawa.

[Mak90] Makino, H., 1990, Versatility index: an indicator for assembly system selection, *CIRP Annals,* 39/1, pp. 15–18.

[Mak93] Makino, H., 1993, Flexible assembly automation makes concurrent engineering possible, *Intl. Conf. on Assembly,* pp. 19–25, Adelaide.

[Mar89] Martens, P., 1989, A product model for automatic assembly planning, *21st Symp. Manufacturing Syst.,* pp. 1–11, Stockholm.

[MAT90] Mattikalli, R.S. et al., 1990, Subassembly identification and motion generation for assembly: a geometric approach, *IEEE Conf. Systems Engineering,* pp. 399–403, Pittsburgh.

[MaY84] Makino, H. and Yamafuji, K., 1984, State-of-the-art of automatic assembly in Japan, *5th Intl. Conf. Assembly Automation,* pp. 1–22, Paris.

[MIC88] Michelini, R.C. et al., 1988, Developments in factory automation: assembly specifications and govern-for-flexibility attainments, *3rd Conf. Computer Aided Production Engineering,* pp. 131–149, Ann Arbor.

[MIC89] Michelini, R.C., 1989, Application of artificial intelligence tools in simulational programming facilities for factory automation, *EERP Survey Report,* DEC Woods Meeting, pp. 801–867, Stresa.

[MIC90] Michelini, R.C. et al., 1990, Integrated management of concurrent shopfloor operations, *Intl. J. Computer-Integrated Manufacturing Systems,* vol. 3, n. 1, pp. 27–37.

[MIC92a] Michelini, R.C. et al., 1992, Simulation facilities for the development of computer-integrated manufacturing, *Intl. J. of Advanced Manufacturing Technology,* vol. 7, n. 4, pp. 238–250.

[MIC92b] Michelini, R.C. et al., 1992, Case studies in modelling the control-and-recovery strategies of robotized manufacturing cells, *1st Worldwide Gensym Users Group Meeting,* pp. 50–65, Santa Margherita.

[MIC92c] Michelini, R.C., 1992, Decision anthropocentric manifold for flexible-specialization manufacturing, *4th Japan-USA Intl. Symposium on Flexible Automation,* vol. 1, pp. 467–474, San Francisco.

[MIC92d] Michelini, R.C. et al., 1992, An expert simulator for factory automation, in *Human Aspects in Computer-Integrated Manufacturing,* G.J. Olling, F. Kimura, Eds., North-Holland, pp. 797–804.

© 2001 by CRC Press LLC

[MIC94a] Michelini, R.C. and Kovacs, G., 1994, Knowledge organization in intelligent manufacturing, *3rd Intl. Workshop on Cooperative Knowledge Processing for Engineering Problem Solving,* Rovereto.
[MIC94b] Michelini, R.C. et al., 1994, Expert scheduler for adaptive planning of batch-varying highly-diversified products, *5th Intl. ASME-ISCIE Symp. on Flexible Automation,* pp. 752–756, Kobe.
[MIC94c] Michelini, R.C. et al., 1994, Flexible manufacturing with integrated control and management, *Modern Manufacturing: Control & Technology,* Zaremba & Prasad, Eds., Springer Verlag, pp. 225–252.
[MIC94d] Michelini, R.C. et al., 1994, Lean engineering and performace measurement in flexible manufacturing, *27th ISATA on Lean/Agile Manufacturing,* pp. 279–286, Aachen.
[MIC95a] Michelini, R.C. et al., 1995, Assessing and exploiting quality control, Workshop 'Quality '95,' Piest'any, pp. 115–120.
[MIC95b] Michelini, R.C. et al., 1995, Knowledge-based emulation for flexible-automation manufacturing, *Intl. Congress Eurosim '95,* pp. 1259–1264, Vienna.
[MIC96a] Michelini, R.C. et al., 1996, Integrated product-and-process govern of robotic assembly cells, *27th Intl. Symp. on Industrial Robots,* pp. 415–420, Milan.
[MIC96b] Michelini, R.C. et al., 1996, Robots with uncertainty and intelligent automation, *2nd ECPD Intl. Conf. Advanced Robotics and Intelligent Automation,* pp. 31–39, Wien.
[MIC97] Michelini, R.C. et al., 1997, Shop controller-manager in intelligent manufacturing, in *Management and Control of Manufacturing Systems,* S. Tzafestas, Ed., Springer Verlag, pp. 218–254.
[MiH89] Miller, J.M. and Hoffman, R.L., 1989, Automatic assembly planning with fasteners, *IEEE Conf. Robotics and Automation,* pp. 69–74, Scottsdale.
[MiH93] Minzu, V., and Henrioud, J.M., 1993, Systematic method for the design of flexible assembly systems, *IEEE Conf. Robotics and Automation,* vol 3, pp. 56–62, Atlanta.
[MIL93] Mills, J.J. et al., 1993, The dynamically reconfigurable assembly system: application issues, *Intl. Conf. on Assembly,* pp. 27–34, Adelaide.
[Mil96] Milutinovic, D., 1996, Micro SCARA as universal compliant wrist, *CIRP Annals,* 45/1, pp. 15–18.
[MiM93] Milutinovic, D. and Makino, H., 1993, A new robotic mechanism based on the SCARA concept, *24th Intl. Symp. Industrial Robots,* pp. 385–390.
[MiR96] Michelini, R.C. and Razzoli, R.P., 1996, Concurrent enterprise paradigms for integrated design and intelligent manufacturing, *Intl. ESoCE Conf. Concurrent Enterprise,* IMAGE-R. 1187, pp. 1–12. Milan.
[MiS90] Milberg, J. and Schmidt, M., 1990, Flexible assembly systems: opportunities and challenge for economic production, *CIRP Annals,* 39/1, pp. 5–8.
[MIY86] Miyakawa, S. et al., 1986, The Hitachi assemblability evaluation method, *1st Conf. Design for Assembly.*
[MIY88a] Miyakawa, S. et al., 1988, The Hitachi assemblability evaluation method and its applications, *Journées de Microtecniques,* pp. 99–114, Lausanne.
[MIY88b] Miyazaki, H. et al., 1988, Toyota group assembly automation experience, *Conf. Proc. IFS: Developments in Assembly Automation,* pp. 39–62, London.
[MoH87] Morris, G.H. and Haynes, L.S., 1987, Robotic assembly by constraints, *IEEE Conf. Robotics and Automation,* pp. 1481–1486, Raleigh.
[Moi86] Moinet, M., 1986, New dimension in automation, *7th Conf. Assembly Automation,* pp. 233–241, Zurich.
[Moi88] Moinet, M., 1988, Practical example of a computer integrated assembly system built up step-by-step, *Conf. Proc. IFS: Developments in Assembly Automation,* pp. 321–328, London.
[MOL92] Molfino, R.M. et al., 1992, Sistema esperto in tempo reale per il governo di un reparto di assemblaggio, *Seminario SIRI-UCIMU Il Montaggio Automatico,* pp. 8.1–16, Milano.
[Mul87] Mullineaux, G., 1987, Optimisation scheme for assembly components, *Computer-Aided Design,* vol. 19, n. 1, pp. 35–40.
[MUS86] Musselman, K.J. et al., 1986, Applying simulation to assembly line design, *Intl. Symp. Simulation,* IFS, pp. 21–34.

© 2001 by CRC Press LLC

[MuY90] Murayama, Y. and Yamazaki, Y., 1990, Automatic assembly of many types of watches, *11th Intl. Conf. Assembly Automation,* MS90-827-1-8, Dearborn.

[NAI93] Naitoh, T. et al., 1993, The development of intelligent body assembly system, *24th Intl. Symp. on Industrial Robots,* pp. 129–136, Tokyo.

[Nar81] Naruki, K., 1981, How to promote automation in Japanese industries: high-speed and flexible automated assembly line, *2nd Intl. Conf. Assembly Automation,* pp. 143–151, Brighton.

[NeW78] Nevins, J.L. and Whitney, 1978, Computer controlled assembly, *Sci. American,* vol. 238, n. 2, pp. 62–74.

[NoR90] Nof, S.Y. and Rajan, V.N., 1990, Planning collaborative automation for flexible assembly, *11th Intl. Conf. Assembly Automation,* MS90-841-1-24, Dearborn.

[OHA83] Ohashi, T. et al., 1983, The automatic assembly line for VTR mechanisms, *4th Intl. Conf. Assembly Automation,* pp. 350–361, Tokyo.

[Oka93] Okano, A., 1993, Computer-aided assembly process planning with resource assignement, *IEEE Conf. Robotics and Automation,* vol. 2, pp. 301–306, Atlanta.

[OnL96] Onori, M. and Langbeck, B., 1996, A new flexible automatic assembly cell for small to medium size product volumes, *27th Intl. Symp. on Industrial Robots,* pp. 551–556, Milan.

[ONO93] Onori, M. et al., 1993, Application of a new programming environment within the MARK III hybrid flexible automatic assembly system, *24th Intl. Symp. Industrial Robots,* Tokyo.

[ONO94a] Onori, M. et al., 1994, Practical implementation of flexible assembly control environment, *25th Intl. Symp. Industrial Robots,* Hannover.

[ONO94b] Onozuka, M. et al., 1994, Computer aided evaluation of assembly sequences based on the analysis of method-time measurement and design for assembly, *Int. J. Advan. of Intelligent Production,* pp. 707–712.

[PaA94] Park, J.H. and Asada, H., 1994, Sequence optimisation for high speed robotic assembly using simulated annealing, *IEEE Conf. Robotics and Automation,* pp. 3441–3446.

[Rog93] Rogers, G.G., 1993, Modular production systems: a concurrent manufacturing philosophy, *IEEE Conf. Robotics and Automation,* pp. 50–55.

[RoL87] Rocheleau, D.N. and Lee, K., 1987, System for interactive assembly modelling, *Computer-Aided Design,* vol. 19, n. 2, pp. 65–72.

[ROT89] Roth, N. et al., 1989, Model generation for sensor-guided flexible assembly systems, *CIRP Annals,* 38/1, pp. 5–8.

[SaD92] Santochi, M. and Dini, G., 1992, Computer-aided planning of assembly operations: the selection of the assembly sequences, *Int. J. Robotics and CIM*, vol. 9, n. 6, pp. 439–446.

[SaH87] Sanderson, A.C. and Homem de Mello, L.S., 1987, Task planning and control synthesis for flexible assembly systems, in *Machine Intelligence and Knowledge Engineering for Robotic Applications,* pp. 331–353.

[Sah93] Sahara, Y., 1993, Flexible parts feeding system using image processing technology, *Robot,* 92, pp. 33–39 (in Japanese).

[SAK96] Sakakibara, S. et al., 1996, An innovative automatic assembly system where a two-armed intelligent robots build mini-robots, *27th Intl. Symp. on Industrial Robots,* pp. 937–942, Milan.

[SAN95] Santochi, M. et al., 1995, Assembly planning software systems, *CIRP Annals,* 44/2, pp. 651–658.

[Sch83] Schaefer, J.O., 1983, Discussion and analysis of conventional part processing vs. a single arm robot-fed system, *4th Intl. Conf. Assembly Automation,* pp. 325–332, Tokyo.

[Sch86] Schmidt, M., 1986, Flexibility in type handling using automatic guided vehicles in assembly, *7th Intl. Conf. Assembly Automation,* pp. 251–262.

[SCH90] Schraft, R.D. et al., 1990, Flexible production automation needs flexible automatic material handling systems, *21st Intl. Symp. on Industrial Robots,* pp. 347–354, Copenhagen.

[SEK83] Sekigushi, H. et al., 1983, Study on automatic determination of assembly sequences, *CIRP Annals,* 32/1, pp. 371–374.

© 2001 by CRC Press LLC

[SEL87] Seliger, G. et al., 1987, Set-up flexible assembly system, *8th Intl. Conf. on Assembly Automation,* pp. 289–301, Copenhagen.
[SEL92] Seliger, G. et al., 1992, Integrated information modelling for simultaneous assembly planning, *8th CIM-Europe Annual,* Conf., Birmingham.
[SEL94] Seliger, G. et al., 1994, *Integration of Recycling: Considerations into Products Design*, Kluwer, New York.
[SHI95] Shiroyama, K. et al., 1995, Development of an assembly productivity system, *Spring Meeting of Jap. Soc. for Precision Engineering,* pp. 833–834 (in Japanese).
[Soh92] Sohlenius, G., 1992, Concurrent engineering, *CIRP Annals,* 41/2, pp. 645–655.
[Son90] Soni, A.H., 1990, Research needs and opportunities in flexible assembly systems, *ASME 2nd Conf. in Flexible Assembly Systems,* pp. 221–228, Chicago.
[StB93] Storm, T. and Boneschanscher, N., 1993, Internal transport: a key problem in flexible cells, *Intl. Conf. on Assembly,* pp. 183–189, Adelaide.
[TaA81] Takahashi, N. and Aoki, J., 1981, Self-contained station module as a new concept for automatic assembly line, *2nd Intl. Conf. Assembly Automation,* pp. 265–276.
[Tak91] Takamori, K., 1991, The CIM system where computers and robots make robots, *Robot,* 83, pp. 77–83 (in Japanese).
[TAM93] Tamaki, K. et al., 1993, A restructurable assembly center employing mobile DD-robots, *24th Intl. Symp. on Industrial Robots,* pp. 119–126, Tokyo.
[Tan83] Taniguchi, N., 1983, Present state of the arts of system design on automated assembly in Japan, *4th Intl. Conf. Assembly Automation,* pp. 1–14, Tokyo.
[Til93] Tillema, H.G., Ed., 1993, MART: an overview of the mobile autonomous robot Twente project, *WA-315,* Feb. 1993, Universiteit Twente.
[Tip69] Tipping, W.V., 1969, *An Introduction to Mechanical Assembly,* Business Books Ltd., London.
[TOB93] Tobita, H. et al., 1993, Development of lean automation equipment for automobile final assembly line, *26th ISATA,* 93LM029, pp. 77–84.
[TUR87] Turner, J.U. et al., 1987, Tolerance analysis in a solid modelling environment, *ASME Conf. Computers in Engineering,* New York.
[UNO84] Uno, M. et al., 1984, Multiple parts assembly robot station with visual sensor, *5th Intl. Conf. Assembly Automation,* pp. 55–64, Paris.
[Val85] Valade, J.M., 1985, Geometric reasoning and automatic synthesis of assembly trajectory, *Intl. Conf. Advanced Robotics,* pp. 43–50, Tokyo.
[Van90] Van Brussel, H., 1990, Planning and scheduling of assembly systems, *CIRP Annals,* 39/2, pp. 637–644.
[VaV89] Van Brussel, H. and Valckenaers, P., 1989, Hierarchical control of a generic flexible assembly cell, *CIRP Annals,* 38/1, pp. 33–36.
[VOH92] Voho, P. et al., 1992, Flexible production system for small batch final assembly, *23rd Intl. Symp. on Industrial Robots,* pp. 159–165, Barcelona.
[WAA92] Waarts, J.J. et al., 1992, A semi-automatic assembly sequence planner, *IEEE Conf. Robotics and Automation,* pp. 2431–2438, Nice.
[WAR87] Warnecke, H.J. et al., 1987, Computer-aided planning of assembly systems, *8th Intl. Conf. Assembly Automation,* pp. 53–65, Copenhagen.
[WaW82] Warnecke, H.J. and Walther, J., 1982, Automatic assembly: state-of-the-art, *3rd Intl. Conf. Assembly Automation,* pp. 1–14, Boeblingen.
[Wie89] Wiendahl, H.P., 1989, Shortening the manufacturing cycle of products with many variants by simultaneous assembly engineering, *10th Intl. Conf. Assembly Automation,* pp. 11–29, Kanazawa.
[WIE91] Wiendahl, H.P. et al., 1991, Modeling and simulation of assembly system, in *CIRP Annals,* 40/2, pp. 577–585.

© 2001 by CRC Press LLC

[Wil90] Wilson, R.H., 1990, Efficiently partitioning an assembly, *IASTED Intl. Conf. Robotics and Manufacturing.*

[Wol90] Wolter, J., 1990, A constrait-based approach to planning with sub-assemblies, *IEEE Conf. Systems Engineering,* pp. 412–415.

[ZUS94] Zussman, E. et al., 1994, Disassembly-oriented assessment methodology to support design for recycling, *CIRP Annals,* 43/1.

© 2001 by CRC Press LLC

3

Multi-Level Decision Making for Process Planning in Computer-Integrated Manufacturing (CIM) Systems

3.1 Introduction

3.2 Conventional Approaches to Process Planning
The Variant Approach • The Generative Approach

3.3 Description of Process Planning Problems
Company Specific and Application Oriented • Time Dependence • Reactive Process Planning • Alternative Process Plans • Uncertainty • The Critiques on Problems of Decision Making

3.4 Manufacturing Processes and In-Process Part Features
The In-Process Part Features • The Abstraction of Manufacturing Processes • The Relationships between In-Process Features and Processes

3.5 The Part State Tree

3.6 Multi-Level Decision Making Based on Artificial Intelligence
Multi-Level Decision Making Using Breadth First Search Method • Multi-Level Decision Making Using Depth First Search Method • Multi-Level Decision Making Using Hill Climbing Search Method • Multi-Level Decision Making Using Least Cost Search Method • Multi-Level Decision Making Using Path Removal Search Method • Multi-Level Decision Making Using Node Removal Search Method • Multi-Level Decision Making toward Optimal Process Planning

3.7 Multi-Level Decision Making Based on Fuzzy Set Theory
Description of a Formal Fuzzy State Model • Multi-Level Fuzzy Decision Making Based on the Fuzzy State Model • Fuzzy Function, Fuzzy Goals, and Fuzzy Constraints • Fuzzy Function $S_{t+1} = f(S_t, P_t)$ • Fuzzy Goals G • Fuzzy Constraints C

Zhengxu Zhao
University of Derby

© 2001 by CRC Press LLC

3.8 Process Planning with Multi-Level Fuzzy Decision Making
3.9 Conclusions

3.1 Introduction

Process planning was traditionally considered as manufacturing preparation that provides manufacturing methods and operation instructions. When computer aided process planning (CAPP) was first attempted in the early 1960s [Neibel, 1965; Schenk, 1966], it emerged as a vital link between computer aided design (CAD) and computer aided manufacturing (CAM). Today process planning has become an important part in computer integrated manufacturing (CIM) environment [Zhang and Alting, 1993; Larsen, 1993; Gulesin and Jones, 1994].

As a formal definition, computer aided process planning is the systematic determination of manufacturing methods and operation details by which parts can be produced economically and efficiently from raw materials to finished products. Since CAPP had played a significant role in CIM and helped companies in increasing productivity and gaining competitiveness, there was a great excitement about CAPP research and, thus, it spawned considerable CAPP system development in both academic community and industrial world. The last two decades saw a proliferation of research publications and system reports, addressing various problems and offering a wealth of different solutions. An extensive literature survey made by Alting and Zhang [1989] covered the state-of-the-art of process planning and most of CAPP systems worldwide at that time, including research and industrial prototypes and commercial packages. Since then, agreement on process planning approaches and techniques have been achieved as being the variant approach based on group technology (GT) and the generative approach based on decision trees, decision tables, logic formulae, knowledge bases, and expert systems [Chang and Wysk, 1985], [Alting and Zhang, 1989], [Gupta, 1990].

However, due to the fast changes in market demands and the influence of new computing technology, manufacturing enterprises are facing increased competition in the dynamic global market. Companies have to respond fast to the market changes in order to succeed in the competition world. Their production has to be flexible with short lead-time and high productivity.

To increase flexibility in a production life cycle, process planning has to play a significant role by dealing with dynamic activities and time-dependent problems from product design to shop floor manufacturing. On the one hand, it has to provide multiple decisions and alternative information transfer from design to various manufacturing functions. On the other hand it must be capable of coordinating, harmonising and integrating production activities such as design, production planning, resource planning, shop floor manufacturing, and controls. To date, however, no existing CAPP systems have ever met such demands. Most of the CAPP systems in use have not gained anticipated computing support and flexible planning functions and tools [Bhaskaran, 1990], [Larsen, 1993], [Zhao and Baines, 1994], [Zhao, 1995], and [Maropoulos, 1995].

The text in this chapter is intended to cover the most recent development and the problems in CAPP and to provide possible solutions to some of those problems. The chapter contains eight sections starting with this current introductory section. The next section briefly describes the conventional approaches to process planning and the techniques involved. The third section highlights the process planning problems that conventional planning techniques have failed to resolve. In the fourth section, manufacturing processes and in-process part features are defined in process planning terms. The generic relationships between manufacturing processes and in-process features are described. Based on those relationships, the concept of part states is derived in the fifth section. A part state tree is built as a process planning solution domain to support effectively most artificial intelligence based multi-level decision making algorithms including fuzzy decision making technique. Section 3.6 describes the implementation of various artificial intelligence (AI) based multi-level decision making algorithms based on the part state tree and shows how those algorithms and the part state tree can be combined to form useful process planning tools. It also shows how process planning knowledge bases can be developed based on the part state tree. Sections 3.7

© 2001 by CRC Press LLC

and 3.8 provides a detailed description of the multi-level fuzzy decision making technique based on a fuzzy part state model. It shows how the technique can deal with problems associating with alternative process plans, uncertain decision making and time related dynamic changes. Finally in section 3.9, useful conclusions are drawn in relating to the present research and future work in the area. Particular discussions are conducted on the multi-level fuzzy decision making techniques and the issues on creation of the part state tree. The literature referenced throughout the text are listed in the end of the chapter.

3.2 Conventional Approaches to Process Planning

The approaches to process planning actually refers to the approaches to the design of CAPP systems (which are mainly computer software packages). Generally there are two conventional approaches, the variant approach and the generative approach. CAPP systems designed with these two approaches are accordingly classified into two categories as variant systems and generative systems.

The Variant Approach

The underlying technology of the variant approach is GT. The variant approach itself can be explained by examining how a variant system is constructed and how the system works. Typically, a variant system is constructed in a way like this.

First, a number of different parts are selected and classified, using a part classification system, into different part families. Then each part in a part family is represented in GT-like code and the part family is represented in a family matrix. The family matrix is supposed to represent all the design and manufacturing features that belong to all the parts in that family. Finally, to each family matrix, there is one (or more than one) predesigned process plan often called master plan attached. Both the master plan and the family matrix are stored in a data base.

When the manufacturing processes for a specific part is to be planned, the part is first defined in the GT-like code. This code is then compared with the family matrices in the data base. If the part code matches a family matrix stored in the data base, the part is then considered as a member of the part family represented by that family matrix. Therefore the master plan attached to that family matrix is retrieved from the data base and is considered to be the process plan for that specific part. Because the parts in a part family are only similar to each other, the process plan retrieved from the data base may not exactly the plan for that specific part. It is only a variation of the actually required plan. Very often, modification on this plan is needed before it can be used in shop floor for actual manufacturing.

Compared with the generative approach described next, the variant approach is a well established approach in terms of the planning techniques and the discipline involved in designing the software tools (mostly being data base management systems). However, nearly all variant systems are virtually databases where both part families and process plans are prepared and stored in advance. The system cannot produce process plans for those parts that do not belong to any of the part families stored in the data base. Besides, creating, updating, and maintaining such a data base can be difficult and costly. For manufacturing processes of discrete products, variant systems offer little practical use. Since most process planning tasks are application-oriented and company specific, variant systems, with little flexibility, are generally not suitable for today's manufacturing applications.

The Generative Approach

The generative approach attempts to overcome the disadvantage of the variant approach by using logic, rules, and decision making algorithms to make creative planning. Generative systems attempt to generate process plans by computerising the knowledge and expertise of a human planner and emulate his or her decision-making process. Although the idea is simple and promising, the techniques developed so far to implement the generative approach is far from adequate to build a practically useful generative system.

© 2001 by CRC Press LLC

The reason for this is the problems which will be discussed shortly in next section. Typical techniques have been available for designing generative systems are those such as decision tables, decision trees, rule bases, artificial intelligence, and expert systems. Although the earlier optimistic speculation was made by Chang and Wysk [1985] on generative systems, most industrial CAPP systems and commercial packages are still developed as being variant or semigenerative. Unless the fundamental process planning problems are fully understood and radical solutions are provided, research and development efforts on existing planning techniques will retain its present form. Additional work along the same lines will be saturated and of little novelty and generic value [Maropoulos, 1995]. This is because process planning is knowledge intensive in nature, which deters planning functions from receiving adequate computing support. More importantly, there is confusion in identification, development, and clustering of software techniques around the planning activities that are involved in uncertain decision making, fuzzy knowledge, and empirical information. Those problem areas are described in more detail below.

3.3 Description of Process Planning Problems

The nature of process planning can be generally described as knowledge-intensive [Kusiak, 1991], [Mill et al., 1993]. The knowledge involved is mostly subjective, nondeterministic, nonheuristic, and difficult to represent. The information handled by various planning functions is often imprecise and vague. The problems inherited from such nature are difficult to resolve using the conventional planning techniques. Below are the generalised descriptions of those problems. Later in the following sections, some of those problems are dealt with multi-level decision-making algorithms developed from AI searching techniques and fuzzy set theory. Others may be attempted with CAPP framework [Zhao, 1997], [Zhao and Baines, 1996 (a)] which is not covered here. The rest could temporarily remain to rely on manual planning and human decision making.

Company Specific and Application Oriented

This is perhaps the most difficult problem that deters CAPP systems from receiving generic and automatic functions. Different companies, different factories, and different applications use different planning data, planning rules, and planning methods.

In variant systems, part families and master plans can only be defined according to particular manufacturing environment and individual applications. In generative systems, the planning knowledge and decision-making rules are defined and set up according to individual planning situations. A universal CAPP system that could be used for different applications is extremely hard to build with current technology. Built-in (hard coded) planning functions and planning tools found in the early CAPP systems are virtually of no practical use for today's manufacturing tasks. Therefore designing flexible and adaptable CAPP systems should be the major concern at present and in the near future.

A special methodology [Zhao, 1997] has been developed toward this problem, where an effective CAPP framework provides users with customised planning tools that can be selected for specific use. A runtime shell that is created to host those customised planning tools and the user-machine interface utilities that support interactive knowledge acquisition and knowledge representation. The description of this methodology is beyond the scope of this chapter. It should be pointed out, however, that the major difficulties for designing a flexible and adaptable CAPP system are resident with acquisition, representation, and maintenance of process planning knowledge. The part state tree and the fuzzy state model presented later in this chapter will provide one possible way of overcoming those difficulties. Another issue relevant to those difficulties is the standardisation of process plans. Details in this topic can be found in the work by ISO 10303-1; STEP Part 1 [1992], Bryan and Steven [1991], Lee, Wysk, and Smith [1995], Jasthi, Rao, and Tewari [1995], and Zhao and Baines [1996 (b)].

© 2001 by CRC Press LLC

Time Dependence

Process planning is time-dependent and dynamic [Larsen, 1991, 1993]. Due to the fact that materials and manufacturing requirements can be altered or changed consecutively through a sequence of manufacturing processes, decision making in every planning stage deals with different dynamic factors.

Taking metal-cutting processes, for instance, where an initial metal block is machined into the finished part, part features with different attributes such as geometry, dimensions, and tolerances are being transformed from one state to another until the part is finally manufactured. To carry out process planning for such processes, it should be done by following a series of dynamic part states. Because the part is manufactured by individual machining processes from one state to another, a sequence of machining processes will transform the part from the initial state (metal block) through different intermediate states (the workpieces) to the final state (the finished part specified in the design or CAD model). As illustrated in Figure 3.1, in order to create a simple process plan that contains machining processes from stage (1) to stage (6), six consecutive part states have to be defined according to the time sequence in which they are being manufactured.

Because the design information inputted to the planning system normally comes only from the finished part state, the definition of the intermediate states of the material must happen within the planning system. The conclusion drawn from this observation is that future CAPP systems should be equipped with sufficient CAD modelling or design functions to generate the information about those intermediate part states.

Reactive Process Planning

The planning functions being capable of dealing with dynamic changes in manufacturing processes and manufacturing requirements have far-fetching importance in modern manufacturing environment. As the development in such areas as open manufacturing systems and shop floor control architectures, process planning is demanded to provide not only off-line information but also on-line data to those manufacturing and control environs. Now on-line planning (in other words, reactive planning or adaptive planning) has already emerged as a practical demand. How future CAPP systems could be equipped with new planning functions to meet with such a demand is a new challenging problem [Lee, Wysk, and Smith, 1995], [Zhao and Baines, 1996 (b)]. So far little research work in relating to this problem has been reported in the literature.

Alternative Process Plans

In one aspect, due to the possibility of having alternative processes, alternative machines, and alternative tools for manufacturing the same part, the same process planning problem could often have alternative solutions [Zhang and Huang, 1994, 1995], [Gupta, 1990]. For example, for machining the same part, alternative machining routes can be used due to the fact that alternative machining operations, machine

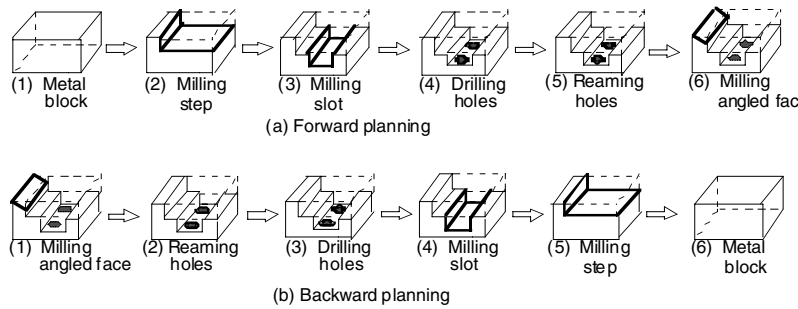


FIGURE 3.1 Machining routes defined as a sequence of part states.

© 2001 by CRC Press LLC

tools, cutting tools, and set-ups could be involved in each machining stage. Thus there can be alternative process plans for manufacturing the same part.

In the other aspect, manufacturing processes involve continuous violation and adjustment to specific prerequisite. The changes of manufacturing circumstances are inevitable and become more frequent. It is desirable for a CAPP system to provide immediately alternative solutions when manufacturing conditions are changed, for example, a machine breakdown. Therefore, generating alternative process plans is an important task for process planning.

The major argument at present is that considering all alternative process plans will poses a combinatorially explosive problem [Bhaskaran, 1990]. Selecting an appropriate plan can be reasonably easy for a human process planner by a trial-and-error method, but it can be difficult for a computer programme using deterministic and heuristic decision making. Conventional generative systems with built-in logic and rules have to ignore this problem by providing single (or limited number of) process plan for each part to be manufactured. CAPP systems based on expert systems or AI techniques promise to provide better solutions for the problem, but such systems require heuristics to support the decision making. Most of those heuristics are local to certain applications and are thus hard to specify and maintain with generic computing methods [Chang, 1990], [Gupta, 1990].

Alternative process plans have become one of the major process planning problems and have received considerable attention. Readers who are interested in this area can refer to articles dedicated to this particular problem. Kusiak and Finke [1988] developed a model, based on minimum cost of machining and minimum number of machines and tools, to select a set of process plans. Bhaskaran [1990] uses a different model which considers more factors such as flow rate of parts, processing time and process steps. A latest attempt on the problem is by Zhang and Huang [1994] who use fuzzy logic to deal with imprecise information and vague knowledge by quantifying the contribution of each process plan to the shop floor performance in terms of fuzzy membership. Using fuzzy set theory to deal with this problem will be particularly explored in Section 3.7 where alternative processes, machines, tools, etc. are employed as constructive alternative planning elements to build fuzzy sets. Based on those fuzzy sets, multi-level decision making is performed among those alternative planning elements.

Uncertainty

As described above, alternative plans resulted from different manufacturing aspects. When decisions are to be made during planning, those aspects will often cause uncertainty.

The first is alternative part features. This can be explained by the example shown in Figure 3.2. Considering three features, the angled face, the slot, and the two holes of the finished part, it is possible that any one of the three features could be selected as the feature to be machined at a particular time.

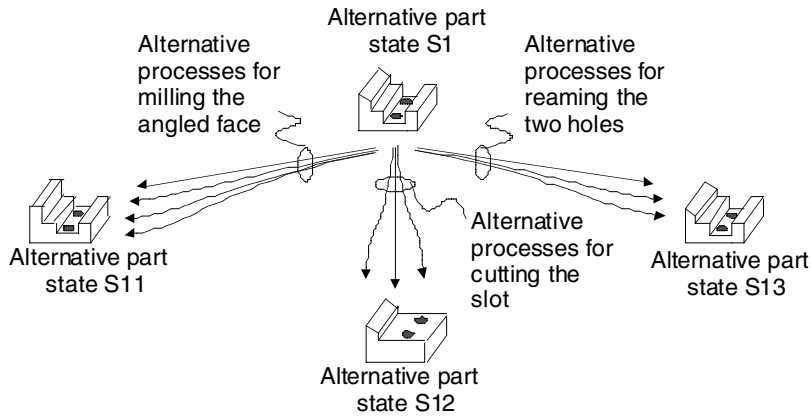

Alternative part state S1
Alternative processes for milling the angled face
Alternative processes for reaming the two holes
Alternative part state S11
Alternative processes for cutting the slot
Alternative part state S13
Alternative part state S12

FIGURE 3.2 Alternative features to be machined.

© 2001 by CRC Press LLC

Therefore, from the finished state (using backward planning), the part could be machined into any one of the three alternative states. It is often uncertain for the computer programme to decide which feature is the most suitable one to be selected and which alternative part state is to be created. In such a dilemmatic situation, conventional CAPP systems have to make a choice arbitrarily or perhaps by relying on users to select using trial-and-error methods.

The second aspect is alternative manufacturing processes which mean that different manufacturing processes could be used to manufacture the part from one specific state to the next specific state. For example, an end face of a shaft can be cut either by face turning or by face milling. Similar to the selection of alternative part features, the selection of alternative manufacturing processes can also be a uncertain decision making process.

The third aspect is alternative machines, tools, operations or set-ups which will result in alternative manufacturing processes and alternative process plans. Again the decision on which machine, tool, operation, and set-up should be used can also be uncertain to make.

In the following sections, alternative machines, tools, operations, and set-ups are used as alternative planning elements to describe alternative manufacturing processes. The decision making techniques presented in those sections consider each manufacturing process as a time interval in which a feature is created or transformed from one state to another by only one manufacturing method (or operation); the manufacturing method is considered as being performed on one machine, with the same type of tools and under one set-up [Zhao, 1995]. With such an arrangement, an alternative machine, an alternative tool, an alternative operation, and an alternative set-up can form an alternative manufacturing process that is unique to be evaluated by specific factors and by fuzzy memberships enforced onto individual processes.

The last aspect is alternative transformed states. This can be explained by an example of a machining process. In a normal case, a machining process can possibly transform a part from a specific state to several alternative states. For instance, a slab milling process can cut a flat face into a cylindrical surface, a conical surface, a curved surface, or another flat surface (remember backward planning is in use here). To decide which surface is actually created after the process, it could be hard to achieve by a computing programme. The possible method is to impose such a condition that the slab milling process creates a surface only by removing the minimum volume of the material from the workpiece. This idea can be expressed in a general way as this. Suppose a machining process P_t can transform a part from a specific state S_t to several different states $S_{t+1,k}$ (where $k = 1, 2, 3, \ldots, N$), the alternative transformed states $S_{t+1,k}$ can be decided by computation of the minimum volume of the removed material. To avoid complicated geometry computation, $S_{t+1,k}$ can be determined manually for individual processes and stored in the data base for use by the decision making programmes.

The Critiques on Problems of Decision Making

The above process planning problems can be generalised in two categories according to the computerised solutions: the computation problems and the decision making problems. Computation problems can always be solved by deterministic procedures or mathematics methods. Those procedures or methods can be easily and successfully implemented by programmes. Unfortunately, only a small portion of such process planning problems fall into this category. The variant approach is effective to deal with such problems by mature techniques like data bases, coding, and classification.

The majority of the problems are decision making problems that include those to be solved with heuristics and those to be solved without heuristics. Heuristic decision-making problems can be solved by search for solutions in predefined knowledge domains guided by given heuristics. Problems without heuristics have to be solved by reasoning that requires high intelligence which at present only human process planners possesses.

Both the heuristic and the nonheuristic problems have the nature of vagueness and uncertainty. Conventional generative approaches, including artificial intelligence based expert systems, to process planning are primarily deterministic and heuristic and are not too concerned with vagueness and uncertainty.

© 2001 by CRC Press LLC

In reality, vagueness and uncertainty are believed to form a large proportion of process planning tasks and have not been well handled by conventional planning approaches. It is therefore not surprising to see that most existing commercial and prototype CAPP systems have to rely on much human intervention whenever nondeterministic problems are encountered. The techniques derived from fuzzy set theory [Zadeh, 1965] for dealing with vagueness and uncertainty have long been available and have had many applications in different fields ranging from medical diagnosis and investment management to consumer electronics and industrial control systems [Mizumoto et al., 1979], [Zadeh, 1991]. Fuzzy set theory aims to providing a body of concepts and techniques for dealing with modes of reasoning which are approximate rather than exact. The objective of fuzzy set is to generalise the notions of a set and propositions to accommodate the type of fuzziness in many decision-making problems. The engineering application of fuzzy set theory has been focused on the area of fuzzy control [Klir and Folger, 1988]. Very little literature is available in applying fuzzy set to process planning [Zhang and Huang, 1994], [Singh and Mohanty, 1991]. The application of expert systems in process planning and the merge of fuzzy set theory with artificial intelligence techniques in other application areas indicates that fuzzy set theory could also provide effective solutions to process planning problems.

Realising the fundamental process planning problems highlighted above, the text below provides effective solutions with useful multi-level decision-making techniques that are derived from artificial intelligence and fuzzy set theory. First, a process planning solution domain is created for computerised multi-level decision making. Second, artificial intelligence based multi-level decision-making algorithms are described and implemented. Third, a multi-level fuzzy decision-making technique is developed. Finally, multi-level process planning decision-making tools developed from the artificial intelligent algorithms and the fuzzy set decision-making techniques are described. As examples, simple process plans are created and presented as decision making results for each techniques.

3.4 Manufacturing Processes and In-Process Part Features

Generally speaking, process planning is a multi-level activity, decision making at one level depends on the decision making at the others. Problems at each level have alternative solutions that cannot be distinctively compared and evaluated; decisions on optimal solutions in individual levels and on optimal process plans at the final level are resident in a domain of alternative manufacturing routes. It therefore suggests that all alternative routes need to be considered if the most suitable one is to be selected. To do so, a multi-level solution domain is needed to support the multi-level decision making. To create such a solution domain, two basic elements are specified, (1) the in-process part features, and (2) the abstraction of manufacturing processes.

The In-Process Part Features

The concept of part features originated in computer automated process planning of machined parts, but the majority of the work seems to be initiated by its applications in computer-aided design (CAD) and computer-aided manufacturing (CAM) [Pratt, 1993], [Case and Gao, 1993]. Part features for process planning are slightly different from those for CAD and CAM applications, they are time-dependent and process-oriented. That features are time-dependent means that the consecutive feature states are formed directly by a sequence of manufacturing processes. For example, features from the initial metal block to the immediate workpieces to the finished part are manufactured in a series of machining processes in different time periods. That features are process-oriented means that features have different behaviours and performances in different processes. For instance, in some cases the geometry and the technical requirements of a particular feature require a process of specific capabilities and, in other cases, the interactions of one feature with other features make some processes impossible due to perhaps tool interference and difficult set-up.

To distinguish them from other features, part features that are time-dependent and process-oriented are called in-process part features or in-process features. In-process features can be defined in terms of

© 2001 by CRC Press LLC

manufacturing methods by relating the features to process functions, process capabilities, and process efficiency (to be discussed shortly).

Geometrically, an in-process feature can be a single surface or a set of related surfaces or a design feature as specified in a CAD model. It can be described by such attributes as geometric form, technical requirements, interaction with other features, spatial position, and orientation during manufacturing. An in-process feature must be unique in terms of manufacturing methods. If one feature needs to be manufactured differently from another feature, the two features are said to be different. For example, the hole machined by drilling and the hole machined by reaming are considered as different in-process features because each has its own tolerances and surface roughness requirements.

Examined within a sequence of manufacturing processes, in-process features can have different states as the initial features, the intermediate features, and the final (or finished) features. The initial features normally belong to the raw materials. The intermediate features are those found in workpieces before or after a manufacturing process. The final features belong to the finished parts as being normally defined in design specifications and part CAD models. By focusing on one manufacturing process, a part is transformed from one state to another. As a result, some of its old in-process features may remain unchanged, others will be transformed and new ones can be created.

The Abstraction of Manufacturing Processes

The word *process* means a procedural course of events or actions that take place in definite manners during a lapse of time, leading to the accomplishment of some results. In process planning, it refers to a time interval during which a course of manufacturing activities or consecutive operations are performed. Here it is specified as such a time interval that contains only one operation that is performed on only one in-process feature. The constituent of a manufacturing process can be abstracted as an input workpiece, an output workpiece, the in-process features, an operation, a machine, and a tool.

The input workpiece and the output workpiece are the two states of the part before and after the process, respectively. The in-process features are geometric entities such as points, lines, curves, and even features of the input and the output workpieces. According to their roles in the process, an in-process feature can be a resulted new feature, a transformed feature, a reference feature (datum), or a clamping feature, see Figure 3.3.

The operation is the action performed on the machine with the tool to change the input workpiece into the output workpiece by following a repertoire of manufacturing instructions. Typically, as in a machining operation, those instructions form a series of cuts (or NC code) described by machining parameters, i.e., cutting speed, cutting feed, and depth of the cuts. The operation is unique. Two operations are the same only when they are performed on the same machine, the same tool with the same manufacturing instructions.

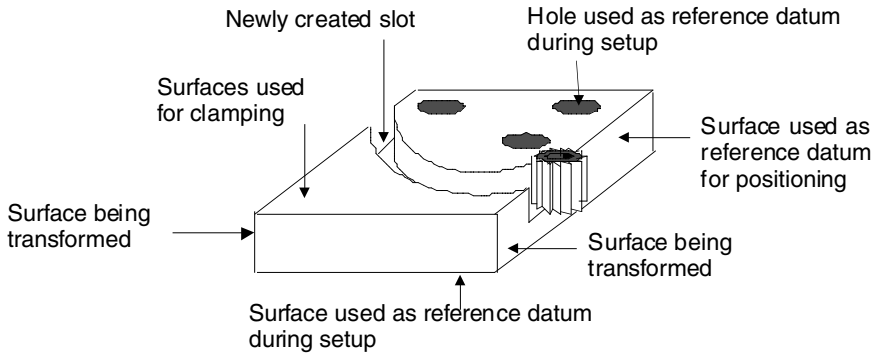

Newly created slot
Hole used as reference datum during setup
Surfaces used for clamping
Surface used as reference datum for positioning
Surface being transformed
Surface being transformed
Surface used as reference datum during setup

FIGURE 3.3 In-process part features within a machining process.

© 2001 by CRC Press LLC

A manufacturing process is evaluated by its function, capability, and efficiency. The function of a process describes the type of the in-process features that it can manufacture. Since the operation within the process is unique, one process can only have one function. Thus, different processes can be identified uniquely by their functions.

Ideally, if the machine and the tool in the process are sufficient in power and precision, all technical requirements like tolerances and dimensions of the workpiece can be taken for granted. The process can then only be concerned with the creation of the geometric form of the in-process features. In reality, every process has a limited range of capability for specific technical requirements. The capability of a process represents the quality of the in-process features that the process is capable to attain such as the attainable dimensions, tolerances, and surface roughness. Process capabilities are mainly decided according to the technical attributes of the input workpiece. For example, a surface roughness of 0.005 μm is a capability of an external cylindrical grinding process, in order to attain this, a cylindrical surface with a roughness of less than 0.05 μm needs to be machined in the previous processes.

According to its function and capability, a process can be selected to manufacture a specific feature. For economic reasons, however, this selected process not only has to manufacture the feature into required form and quality but also to achieve the best economic result such as short manufacturing time and low cost. This requirement is specified as the efficiency of the process. If the process efficiency is evaluated by manufacturing time and cost, it can be calculated and presented quantitatively in a traditional way [Curtis, 1988] by considering such factors as machine and tool capacity, production volume, and overhead cost.

With its function, capability, and efficiency specified as above, a manufacturing process can be defined, selected, and evaluated during process planning in a way like this: first, its function is considered for achieving the specified feature form; next, the capability for attaining the technical requirements is examined; finaly, the efficiency for fulfilling the expected economic results is verified.

The Relationships between In-Process Features and Processes

Most in-process features have regular geometrical forms that can be mathematically described. The operation in a manufacturing process and the geometrical form of an in-process feature within that process allow themselves to be described in the same way. The slot shown in Figure 3.4, for example, can be described in two mathematical ways, each in turn forms the basis of a machining operation for cutting the slot. From this observation, two types of relationships between in-process features and manufacturing processes can be derived, which are described below. More detailed descriptions can be found in publications [Zhao, 1992], [Zhao et al., 1993], and [Zhao and Baines, 1992].

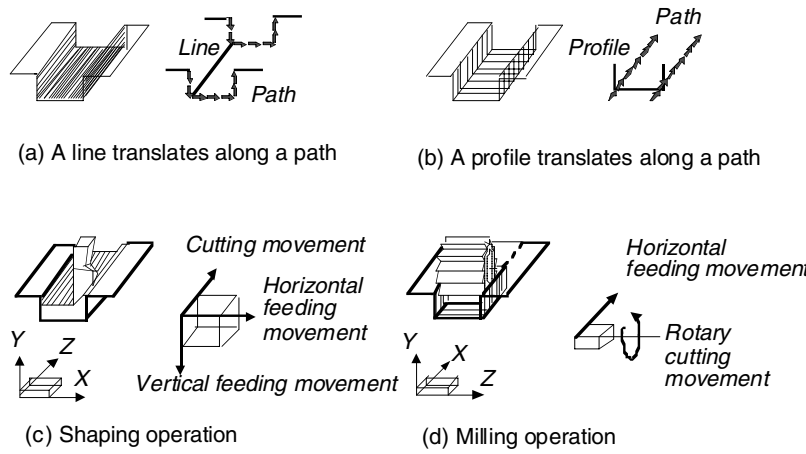


FIGURE 3.4 Descriptions of in-process features and processes.

© 2001 by CRC Press LLC

First, if an in-process feature can be manufactured in a manufacturing process, then the geometric form of that in-process feature and the operation in that manufacturing process must be dual-representable. Both can represent the function of the manufacturing process and both can be described into the same data format. This type of relations has laid the foundation for those decision-making rules such as

$$\text{IF it is feature } F \text{ THEN process } P \text{ is used} \tag{3.1}$$

Second, if a technical attribute (a geometric tolerance, for example) of the in-process feature can be attained by a manufacturing process, the capabilities of that process should always be above the technical requirements of that feature. This forms another category of decision-making rules as follows.

$$\begin{aligned} &\text{IF it is feature } F \text{ of attribute } A1, A2, A3, \ldots, An, \\ &\text{THEN process } P \text{ of capabilities } C1, C2, C3, \ldots, Cn \text{ is used.} \\ &\text{where } A1 \leq C1, A2 \leq C2, A3 \leq C3, \ldots, An \leq Cn. \end{aligned} \tag{3.2}$$

Third, the primary task of a machining process is to generate the geometrical form of a feature. Even if a process is mainly used to obtain specific technical requirements of the feature; its function must be exactly matched to the geometrical form of that feature. Normally, the higher the capability the process has, the easier the technical requirements of the feature can be achieved. However higher capability of process usually leads to longer and more expensive manufacturing. Therefore, given an in-process feature, the manufacturing process to manufacture the feature should be planned in an order like this. The process is first selected according to its function that matches the geometric form of the feature. Then the process is validated by its capability against the technical requirement of the feature. The process is finally evaluated by its efficiency based on maximum manufacturing time and cost that could be possibly allocated to it. For example, a finish turning process must have the capability to cut the cylinder with surface roughness less than specified. To do this, it must have the function of machining a cylindrical surface. It should also be such a process that has the proper capability to avoid machining the cylinder of unnecessarily high surface finish and high cost. This can be generalised into the third category of decision-making rules as

$$\begin{aligned} &\text{IF it is feature } F \text{ of attribute } A1, A2, A3, \ldots, An, \\ &\text{AND there is manufacturing time } T \text{ and cost } M, \\ &\text{THEN process } P \text{ of capabilities } C1, C2, C3, \ldots, Cm, \\ &\text{AND processing time } Tp \text{ and cost } Mp \text{ is used.} \\ &\text{where } Tp \leq T \text{ and } Mp \leq M. \end{aligned} \tag{3.3}$$

3.5 The Part State Tree

A part normally has different in-process features, each feature can be developed through a series of machining processes. If the development of all the features are viewed together, the part would look like an object being transformed consecutively from its initial state to its final state through a sequence of manufacturing processes by changing its shape, size, and technical attributes.

The part may be manufactured from alternative initial states (or raw materials), for example, a blank block or a bar, but it can only have one final state, i.e., the finished part. The states between an initial state and the final state are intermediate states. Between every two consecutive states there exist alternative manufacturing processes. Each of the alternative processes takes an in-process feature as its input from the state near the initial state and transforms or generates a new in-process feature as its output toward the final state.

© 2001 by CRC Press LLC

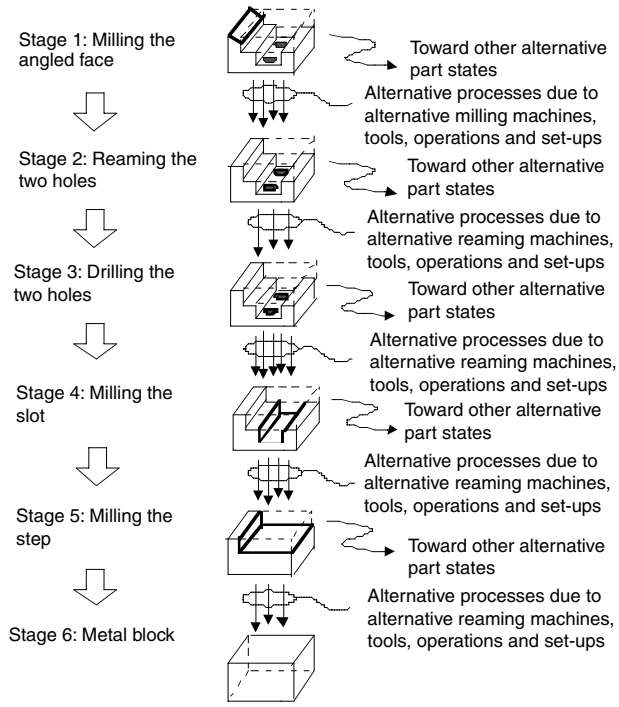

Stage 1: Milling the angled face
Toward other alternative part states
Alternative processes due to alternative milling machines, tools, operations and set-ups
Stage 2: Reaming the two holes
Toward other alternative part states
Alternative processes due to alternative reaming machines, tools, operations and set-ups
Stage 3: Drilling the two holes
Toward other alternative part states
Alternative processes due to alternative reaming machines, tools, operations and set-ups
Stage 4: Milling the slot
Toward other alternative part states
Alternative processes due to alternative reaming machines, tools, operations and set-ups
Stage 5: Milling the step
Toward other alternative part states
Alternative processes due to alternative reaming machines, tools, operations and set-ups
Stage 6: Metal block

FIGURE 3.5 Formation of state paths of machining processes.

In machining processes, features that constitute a finished part are not all created simultaneously at one process, but in a specific sequence of processes. As shown in Figure 3.5, due to the fact that a feature can be machined in different stages and each stage may use alternative processes, the evolution of the part from the initial state to the final state can follow different paths, each consisting of different intermediate states. Those paths are called state paths. Each state path represents a sequence of processes which in turn form a process plan. If all the possible paths are considered, a part state tree can be constructed as shown in Figure 3.6.

The nodes in the tree represent the part states and the arrow lines between every consecutive node represent alternative machining processes. If every state in the tree is labelled according to its position in the tree, the finished part (root node) is *S1*. In the next level, the far left state is *S11*, the middle state is *S12*, and the far right state is *S13*. Those states in the next downward levels can be labelled accordingly, for example, *S111*, *S112*, and *S113*; *S121* and *S122*; *S131*, *S132*, and *S133*. By using such labels, a state path can be identified from the root node to the specified leaf node. For example, the far right path is S_1-S_{13}-S_{133}-S_{1332}-S_{13321}-S_{133211}, which is the path as depicted in Figure 3.5.

A state tree is built based on two facts. The first is that a single part state can be manufactured from alternative in-process features (see Figure 3.2). The second is that a single in-process feature can be manufactured by alternative processes due to alternative machines, tools, operations, and set-ups, for example, the slot can be cut either by shaping or by milling (see Figure 3.4 and Figure 3.5). In Figure 3.6, the arrow lines between every two consecutive states are used only to symbolise the alternative processes; they do not necessarily represent the actual number of the possible alternative processes.

3.6 Multi-Level Decision Making Based on Artificial Intelligence

The part state tree provides a suitable solution domain for AI based multi-level decision-making methods. It makes the implementation of the multi-level decision making simple and easy by using most of the conventional AI searching algorithms. Some of those algorithms are described below to

© 2001 by CRC Press LLC

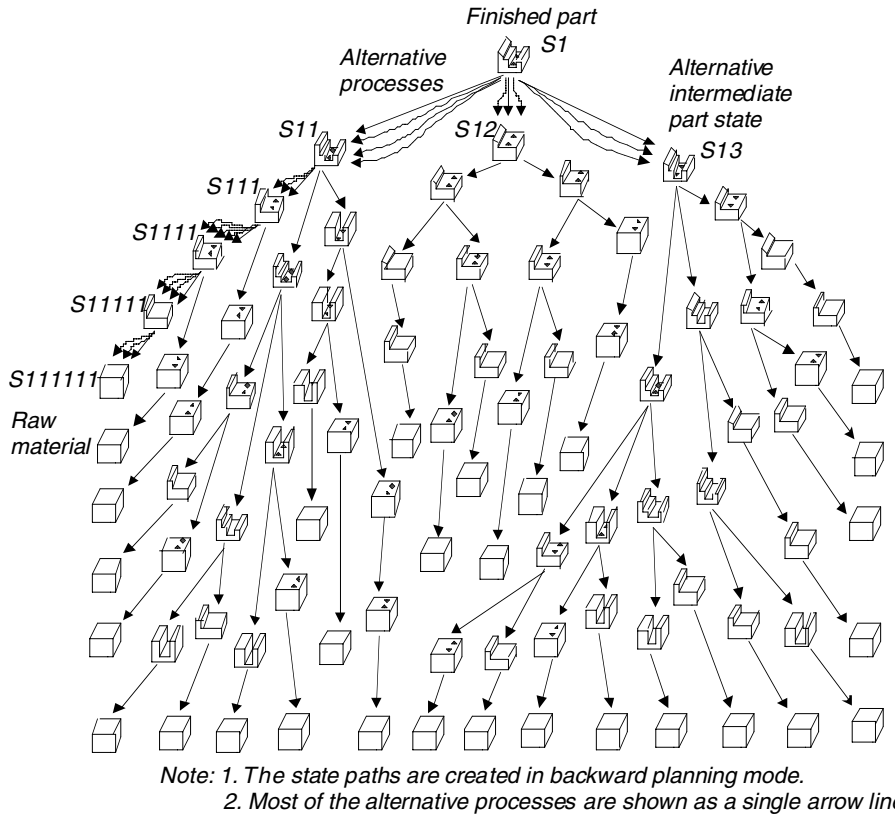


FIGURE 3.6 Part state tree of machining processes.

show how they can be supported by the above part state tree to create useful decision making tools for process planning. But first, the special programming techniques such as object-oriented programming (OOP), data bases, backtracking method, and list-processing method need to be mentioned.

The use of object-oriented programming makes the planning tools to be adopted in different application programmes without changing the source code. Data bases, which are often fashionably called by AI researchers as knowledge bases, are used to hold the part state tree and other planning data. This will enable a decision-making programme to search a part state, a manufacturing process, a machine, a tool, an operation, a set-up method, or a sequence of processes from a large data base, not from a small memory buffer. Therefore the size of the part state tree will not be limited by the computer memory resource. Backtracking, which is basically a stack-oriented operation, provides a routine with a means of finding a solution to specific problems. It allows an artificial intelligence searching algorithm to look for a solution within the part state tree for a planning problem by following various paths of reasoning. If a routine encounters a dead end, then it simply backtracks to an earlier node in the searching process and tries an alternative approach. Lists, which can be designed as character strings, consist of one or more tokens. A token is a programming term that defines the indivisible part of the list. Because tokens may be removed in the order in which they appear in a string, the primary operation will be obtaining the next token, which implies that the token is removed from the list. This technique supports artificial intelligence's traditional list concept as head and tail, with the head of the list being the next token and the tail being what remains on the list. To implement the decision making techniques described, only two basic routines can be employed for list operation. One is to retrieve the next token from the head of the list and the other is to return the current token to the head of the list.

© 2001 by CRC Press LLC

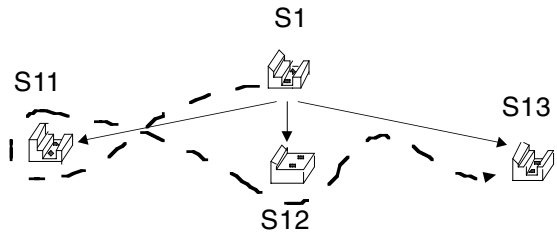


FIGURE 3.7 Breadth first search route.

The decision-making techniques described below use these special programming methods. To limit the length of the text, details on computing programmes and source codes are excluded from the following description.

Multi-Level Decision Making Using Breadth First Search Method

Breadth first search (BFS) algorithm examine every part state on the same level of the part state tree and then proceed to check each state on next level. The search will eventually degenerate into an exhaustive search, therefore the algorithm will always find a solution if one exists in the solution domain. The mechanism of this algorithm can be depicted in Figure 3.7 where the search goal is *S13*. The search will visit nodes *S1*, *S11*, and *S12* before it reaches *S13*. BFS method could reach a solution or sometimes by chance even an optimal solution without backtracking, but it very much depends on the physical organisation of the part state tree. If the search goal is located several levels deep in the state tree, this method could cost substantial search effort to find it.

BFS programme can be designed as a planning tool in a black-box style, this will make it a building block when being used to construct a larger process planning system. It can also be designed as an linkable module called by other applications, for example, a Windows programme. In either case, the part state tree must be first established and then loaded into the data base.

The part state tree must be represented in a form that the search algorithm can easily interpret. The example below shows a typical ASCII text file format that was used by the author to test the process planning tools developed from the techniques described in this chapter. It describes only a state path. The entire state tree can be defined by simply including all the state path in the text file. The file consists of blocks. Each block has a start node and an end node that describe the two consecutive part states in a state path. The link in a block describes the function of the manufacturing process between the two part states. The attribute describes the capability and efficiency of the process (in the example below, efficiency is described with machining time).

```
startnode:   FINISHED PART (S1);
endnode:     ANGLE FACE TO CUT (S11);
link:        MILLING ON MACHINE M1;
attribute:   10;
startnode:   FINISHED PART (S1);
endnode:     SLOT TO CUT (S12);
link:        SLOTTING ON MACHINE M2;
attribute:   20;
startnode:   FINISHED PART (S1);
endnode:     HOLES TO REAM (S13);
link:        REAMING HOLES ON MACHINE M3;
attribute:   25;
```

© 2001 by CRC Press LLC

When any two-part states are specified, for instance, finished part and raw material, the decision-making technique based on breadth first search will search from the state tree for a machining route between the finished part and the raw material. An example of such a route is shown below, which is the far left state path in the part state tree as shown in Figure 3.6. It is generated by a decision-making programme developed from this technique.

Solution is found by BFS to be as follows:

(node) FINISHED PART
by (link) MILLING ON MACHINE M1
with (attribute as) 10 to
(node) ANGLE FACE TO CUT (S11)
by (link) SLOTTING ON MACHINE M2
with (attribute as) 50 to
(node) SLOT TO CUT (S111)
by (link) REAMING HOLES ON MACHINE M3
with (attribute as) 54 to
(node) HOLES TO REAM (S1111)
by (link) DRILLING HOLES ON MACHINE M4
with (attribute as) 22 to
(node) HOLES TO DRILL (S11111)
by (link) MILLING STEP ON MACHINE M2
with (attribute as) 32 to
until finally reaches (node) RAW MATERIAL

Multi-Level Decision Making Using Depth First Search Method

Depth first search (DFS) algorithm is the opposite of BFS. It explores each state path from the root node of the part state tree toward the leaf node in the state path before it tries another state path. Like BFS algorithm, DFS is certain to find a solution if there is one in the part state tree. This is because it will eventually degenerate into an exhaustive search. Figure 3.8 illustrates how this algorithm works. Suppose the search goal is state *S122*. The search will visit the part states *S1*, *S11*, and *S111*. It then backtracks to state *S11*, visit state *S112*, then back to *S11* and *S1*. From state *S1* it searches states *S12* and *S121* and then backtracks to state *S12* before it finally reaches state *S122*.

With the BFS programme being available, implementing the decision-making technique based on DFS can be quite easy when using the object-oriented programming. This is because most of the code for BFS technique can be reused for DFS technique, simply by deriving the DFS function class from the BFS function class. Below is a solution generated by DFS technique from the part state tree.

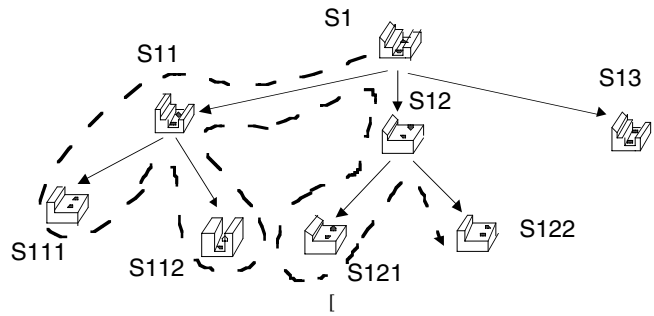


FIGURE 3.8 Depth first search route.

© 2001 by CRC Press LLC

Solution is found by DFS to be as follows:

(node) FINISHED PART
by (link) MILLING ON MACHINE M1
with (attribute as) 10 to
(node) ANGLE FACE TO CUT (S11)
by (link) MILLING STEP ON MACHINE M2
with (attribute as) 20 to
(node) STEP TO CUT (S113)
by (link) REAMING HOLES ON MACHINE M3
with (attribute as) 25 to
(node) HOLES TO REAM (S1131)
by (link) DRILLING HOLES ON MACHINE M4
with (attribute as) 10 to
(node) HOLES TO DRILL (S11311)
by (link) SLOTTING ON MACHINE M2
with (attribute as) 50 to
until finally reaches (node) RAW MATERIAL

Multi-Level Decision Making Using Hill Climbing Search Method

Both BFS and the DFS are blind searching algorithms, i.e., the searching process is not guided by rules. For this reason, the two solutions achieved by the two techniques as being showed above are different.

In most cases, process planning solutions have to be found under certain conditions and constraints. For example, two manufacturing routes *MR1* and *MR2* are both applicable to manufacturing a part, but only the better one can be actually used. If short manufacturing time is a priority, then the manufacturing route involved with fewer machines and set-ups should be selected. Therefore, when searching for the better route, the decision-making programme can use such an assumption that the smaller the number of the machines, operations, and set-ups involves, the shorter the manufacturing time should be. This type of assumption is used as heuristics during the search. Using heuristics to guide a search is in fact maximising or minimising some parameters of the part state tree. Here only two basic heuristic search algorithms are presented. This section describes the hill climbing search (HCS). The next section will describe the least cost search (LCS).

The multi-level decision-making technique developed from the HCS algorithm approaches the final decision step by step. In each step it chooses from the part state tree the part state that is the closest to the decision.

HCS could incidentally reach an optimal solution faster than nonheuristic methods like BFS and DFS. The reason is that it tends to reduce the number of part states that need to be explored before it reaches the decision. However, the technique has three major disadvantages. The first is the plateaus problem that makes the search uncertain especially when all of the part states to be taken in next step seem equally good or bad. In this case, the method is no better than DFS method. The second is the false peak problem that makes the search process backtrack extensively. The third is the ridge problem which makes the search repeat the same part state several times during backtracking.

To implement the technique, all functions in BFS class and DFS class can be derived to form an HCS class by following the object-oriented programming concept. Since the part state tree is constructed with nodes and links and since each link is specified with an attribute, the technique is extremely simple to implement if the search process is guided by the greatest link attribute value. Therefore the HCS technique can be used to search for a manufacturing process (an operation, a machine, a tool, etc.) that has more capability or more efficiency than other processes. It is particularly useful when there are alternative processes between every two part states and each process is evaluated in terms of short manufacturing time, low cost, and high flexibility. For example, by making the link attribute in the state tree represent

© 2001 by CRC Press LLC

the process efficiency, this technique will generate the following machining route from the part state tree as shown in Figure 3.6:

Solution is found by HCS to be as follows:

(node) FINISHED PART
by (link) REAMING HOLES ON MACHINE M3
with (attribute as) 25 to
(node) HOLES TO REAM (S13)
by (link) SLOTTING ON MACHINE M2
with (attribute as) 20 to
(node) SLOT TO CUT (S133)
by (link) DRILLING HOLES ON MACHINE M4
with (attribute as) 25 to
(node) HOLES TO DRILL (S1332)
by (link) MILLING ON MACHINE M1
with (attribute as) 10 to
(node) ANGLE FACE TO CUT (S13321)
by (link) MILLING STEP ON MACHINE M2
with (attribute as) 30 to
until finally reaches (node) RAW MATERIAL

Multi-Level Decision Making Using Least Cost Search Method

This decision-making technique is developed based on another heuristic search algorithm called the least cost search (LCS) which is the opposite to the HCS. LCS has the similar strength and weakness as those of HCS except in reverse order: it has the false valley problem, the lowland problem, and the gorge problem. Like HCS technique, LCS technique can also be implemented by deriving an LCS class from the BFS or DFS class.

The technique using LCS is particularly useful to process planning problems that require the state tree to have the least link attribute value. The least attribute value can represent, for example, the manufacturing time thus the technique can be used to search for a machining route that lead to a shorter manufacturing time. Such a machining route is generated from the part state tree as follows.

Solution is found by LCS to be as follows:

(node) FINISHED PART
by (link) MILLING ON MACHINE M1
with (attribute as) 10 to
(node) ANGLE FACE TO CUT (S11)
by (link) MILLING STEP ON MACHINE M2
with (attribute as) 20 to
(node) STEP TO CUT (S113)
by (link) REAMING HOLES ON MACHINE M3
with (attribute as) 25 to
(node) HOLES TO REAM (S1131)
by (link) DRILLING HOLES ON MACHINE M4
with (attribute as) 10 to
(node) HOLES TO DRILL (S11311)
by (link) SLOTTING ON MACHINE M2
with (attribute as) 50 to
until finally reaches (node) RAW MATERIAL

© 2001 by CRC Press LLC

Multi-Level Decision Making Using Path Removal Search Method

Manufacturing processes are dynamic and change from time to time when manufacturing requirements, company policy, shop floor activities, or technical data are changed or updated. Process planning must react to those changes by providing alternative solutions either as off-line plans or as on-line plans. As mentioned earlier, in off-line planning, alternative solutions are prepared in advance and can be made available when there is a request. In on-line planning, the solutions are generated in real-time whenever a request has been made. Both off-line and on-line planning require planning functions that are able to provide multiple solutions to the same problem.

There are many decision-making algorithms for generating multiple solutions, only two are described below to show how they can be used with the part state tree to develop process planning tools: the path removal search (PRS) method and the node removal search (NRS) method. The former is described in this section and the latter is described in the next section.

The concept of the PRS is simple. When it is searching for a solution, it removes from the part state tree all the part states that constitute that solution. This will make the state tree being updated to allow another decision to be made later. In fact once the PRS method has made a decision, it deletes the part states relating to the decision from the part state tree. The implementation of the method is similar to others described so far. By deriving from the BFS or the DFS class, most of the code for the previously described techniques can be reused here. The PRS can make as many decisions as the part state tree can provide. The following three machining routes are created from the part state tree shown in Figure 3.6:

Solution 1 is found by PRS to be as follows:

(node) FINISHED PART
by (link) MILLING ON MACHINE M1
with (attribute as) 10 to
(node) ANGLE FACE TO CUT (S11)
by (link) MILLING STEP ON MACHINE M2
with (attribute as) 20 to
(node) STEP TO CUT (S113)
by (link) REAMING HOLES ON MACHINE M3
with (attribute as) 25 to
(node) HOLES TO REAM (S1131)
by (link) DRILLING HOLES ON MACHINE M4
with (attribute as) 10 to
(node) HOLES TO DRILL (S11311)
by (link) SLOTTING ON MACHINE M2
with (attribute as) 50 to
until finally reaches (node) RAW MATERIAL

Solution 2 is found by PRS to be as follows:

(node) FINISHED PART
by (link) SLOTTING ON MACHINE M2
with (attribute as) 20 to
(node) SLOT TO CUT (S12)
by (link) REAMING HOLES ON MACHINE M3
with (attribute as) 45 to
(node) HOLES TO REAM (S121)
by (link) DRILLING HOLES ON MACHINE M4
with (attribute as) 25 to
(node) HOLES TO DRILL (S1211)
by (link) MILLING ON MACHINE M1

© 2001 by CRC Press LLC

with (attribute as) 10 to
(node) ANGLE FACE TO CUT (S12111)
by (link) MILLING STEP ON MACHINE M2
with (attribute as) 25 to
until finally reaches (node) RAW MATERIAL

Solution 3 is found by PRS to be as follows:

(node) FINISHED PART
by (link) REAMING HOLES ON MACHINE M3
with (attribute as) 25 to
(node) HOLES TO REAM (S13)
by (link) MILLING ON MACHINE M1
with (attribute as) 10 to
(node) ANGLE FACE TO CUT (S131)
by (link) SLOTTING ON MACHINE M2
with (attribute as) 20 to
(node) SLOT TO CUT (S1311)
by (link) MILLING STEP ON MACHINE M2
with (attribute as) 30 to
(node) STEP TO CUT (S13111)
by (link) DRILLING HOLES ON MACHINE M4
with (attribute as) 30 to
until finally reaches (node) RAW MATERIAL

Multi-Level Decision Making Using Node Removal Search Method

This decision-making technique is developed from another multiple search algorithm called the node removal search (NRS). The NRS generates multiple solutions by removing the last part state in the currently searched state path while it carries on search for another state path. The implementation of the technique is a little more complicated than that of the PRS method because more computing functions are required. The technique can also make as many decisions as the part state tree can provide. However, each time the search process is finished, the part state tree must be recovered to its initial state by repairing the destroyed nodes. As examples, four multiple machining routes produced by this technique are shown as follows:

Solution 1 is found by NRS to be as follows:

(node) FINISHED PART
by (link) MILLING ON MACHINE M1
with (attribute as) 10 to
(node) ANGLE FACE TO CUT (S11)
by (link) MILLING STEP ON MACHINE M2
with (attribute as) 20 to
(node) STEP TO CUT (S113)
by (link) REAMING HOLES ON MACHINE M4
with (attribute as) 25 to
(node) HOLES TO REAM (S1131)
by (link) DRILLING HOLES ON MACHINE M4
with (attribute as) 10 to
(node) HOLES TO DRILL (S11311)
by (link) SLOTTING ON MACHINE M2
with (attribute as) 50 to

© 2001 by CRC Press LLC

until finally reaches (node) RAW MATERIAL

Solution 2 is found by NRS to be as follows:

(node) FINISHED PART
by (link) MILLING ON MACHINE M1
with (attribute as) 10 to
(node) ANGLE FACE TO CUT (S11)
by (link) MILLING STEP ON MACHINE M2
with (attribute as) 20 to
(node) STEP TO CUT (S113)
by (link) REAMING HOLES ON MACHINE M3
with (attribute as) 25 to
(node) HOLES TO REAM (S1131)
by (link) SLOTTING ON MACHINE M2
with (attribute as) 50 to
(node) SLOT TO CUT (S11312)
by (link) DRILLING HOLES ON MACHINE M4
with (attribute as) 30 to
until finally reaches (node) RAW MATERIAL

Solution 3 is found by NRS to be as follows:

(node) FINISHED PART
by (link) MILLING ON MACHINE M1
with (attribute as) 10 to
(node) ANGLE FACE TO CUT (S11)
by (link) MILLING STEP ON MACHINE M2
with (attribute as) 20 to
(node) STEP TO CUT (S113)
by (link) SLOTTING ON MACHINE M2
with (attribute as) 50 to
(node) SLOT TO CUT (S1132)
by (link) REAMING HOLES ON MACHINE M3
with (attribute as) 54 to
(node) HOLES TO REAM (S11321)
by (link) DRILLING HOLES ON MACHINE M4
with (attribute as) 30 to
until finally reaches (node) RAW MATERIAL

Solution 4 is found by NRS to be as follows:

(node) FINISHED PART
by (link) MILLING ON MACHINE M1
with (attribute as) 10 to
(node) ANGLE FACE TO CUT (S11)
by (link) SLOTTING ON MACHINE M2
with (attribute as) 20 to
(node) SLOT TO CUT (S111)
by (link) REAMING HOLES ON MACHINE M3
with (attribute as) 54 to
(node) HOLES TO REAM (S1111)
by (link) DRILLING HOLES ON MACHINE M4

© 2001 by CRC Press LLC

with (attribute as) 22 to
(node) HOLES TO DRILL (S11111)
by (link) MILLING STEP ON MACHINE M2
with (attribute as) 32 to
until finally reaches (node) RAW MATERIAL

Multi-Level Decision Making toward Optimal Process Planning

All of the techniques described above are involved in making single or multiple decisions with or without using heuristics. Ideally, manufacturing processes should be planned to achieve the best results in terms of time, cost, and quality. Although it is difficult to establish optimised process plans, the best solutions to specific planning problems can be achieved. For example, within the part state tree, a machining route that requires minimum manufacturing time can be searched first by using the PRS technique to generate multiple solutions and then by employing the LCS technique to find the route with the minimum manufacturing time. This is the method that is used here. It will also facilitate the implementation process by deriving the code from BFS or DFS functions.

To find the machining route with the shortest time, the programme must retain the currently generated route whose machining time is shorter than the previous route. In this way when the programme ceases making decisions, only the route with the shortest time is left. To keep the current route that has the shorter time, it needs an extra stack file in the knowledge base. Once the decision-making process is finished, this extra stack file will hold the optimal machining route that has the shortest manufacturing time generated from the state tree. As an example, an optimal machining route is created from the part state tree in Figure 3.6 and is shown below.

Optimal solution found from the part state tree is as follows:

(node) FINISHED PART
by (link) MILLING ON MACHINE M1
with (attribute as) 10 to
(node) ANGLE FACE TO CUT (N11)
by (link) MILLING STEP ON MACHINE M2
with (attribute as) 20 to
(node) STEP TO CUT (N113)
by (link) REAMING HOLES ON MACHINE M3
with (attribute as) 25 to
(node) HOLES TO REAM (N1131)
by (link) DRILLING HOLES ON MACHINE M4
with (attribute as) 10 to
(node) HOLES TO DRILL (N11311)
by (link) SLOTTING ON MACHINE M2
with (attribute as) 50 to
until finally reaches (node) RAW MATERIAL

3.7 Multi-Level Decision Making Based on Fuzzy Set Theory

As being addressed earlier, the uncertainty and vagueness of process planning problems can be dealt with fuzzy set theory. In contrast to the techniques just described above, fuzzy set theory deals with approximate reasoning and accommodates fuzzy and uncertain decision-making problems. The following text elaborates the method of using the fuzzy set theory to perform multi-level fuzzy decision making and the technique for implementing the method using the part state tree.

© 2001 by CRC Press LLC

Description of a Formal Fuzzy State Model

In general, any part state tree similar to the one shown in Figure 3.6 represents a multi-level decision-making solution domain that can be expressed as a mathematic model

$$S_{t+1} = f(S_t, P_t) \tag{3.4}$$

where S_{t+1} and S_t are the part states at level t and $t + 1$, respectively, both taking a value from a part state set $X = \{X_1, X_2, X_3,\ldots,X_N\}$; P_t is a manufacturing process at level t and takes a value from a manufacturing process set $Y = \{Y_1, Y_2, Y_3,\ldots,Y_M\}$. X stands for all possible part states to be manufactured; Y is the input set to the decision-making domain, standing for all possible manufacturing processes in a specific manufacturing system. t stands for the level or time and has values 1, 2, 3,…, T; f is a part state transformation function, it can be an ordinary function, a stochastic function, or a fuzzy function. If it is an ordinary function, the part state domain is an ordinary domain like the one used with the AI-based techniques described earlier. If it is a fuzzy function, then the part state domain is a fuzzy domain.

In multi-level fuzzy decision making, a fuzzy decision is evaluated at the last level when $t = T$ based on following fuzzy membership function

$$\mu_{B_T}(S_T) = [0, 1], (S_T \varepsilon B_T) \tag{3.5}$$

where B_T is the fuzzy decision set that satisfies specific fuzzy goals at level $t = T$; S_T is a part state when $t = T$, it can be considered as a fuzzy member of B_T when it represents a manufacturing route; the valuation set [0, 1] is a real type interval, the closer the value of $\mu_{B_T}(S_T)$ is to 1, the more likely S_T belongs to the fuzzy decision set B_T.

For input P_t in expression (3.4) there are a set of fuzzy constraints C_t which is relevant to fuzzy set Y. Therefore, for a series of inputs $P_1, P_2, P_3,\ldots,P_{T-1}$, there is an ideal fuzzy decision set D which is a subset of $Y \times Y \times Y\ldots, Y \times Y$. The fuzzy membership function of D is determined by

$$\begin{aligned}\mu_D(P_1, P_2, P_3,\ldots,P_{T-1}) = {} & \mu_{C1}(P_1)\Lambda\mu_{C2}(P_2)\Lambda\mu_{C3}(P_3)\Lambda\ldots,\Lambda \\ & \mu_{CT-1}(P_{T-1})\Lambda\mu_{B_T}(S_T)\end{aligned} \tag{3.6}$$

where S_T is determined using expression (3.4), i.e., $S_T = f(S_{T-1}, P_{T-1})$.

Assuming that the specific fuzzy goals satisfied by the fuzzy decision set B_T can also be satisfied by the decisions in set D, the ultimate task of decision making is to find out the optimal decisions in set D, i.e., a series of optimal manufacturing processes $P_1, P_2, P_3,\ldots,P_{T-1}$, that has the maximum value of fuzzy membership μ_D.

Multi-Level Fuzzy Decision Making Based on the Fuzzy State Model

Suppose an optimal decision is represented by a series of optimal manufacturing processes $Pmax_1$, $Pmax_2$, $Pmax_3$, …, $Pmax_{T-1}$, then formula (3.6) becomes

$$\begin{aligned}& \mu_D(Pmax_1, Pmax_2, Pmax_3,\ldots,Pmax_{T-1}) \\ = {} & Max_{\{(P1,\,P2,\,P3,\ldots,PT-1)\varepsilon Y\}}\{\mu_{C1}(P_1)\Lambda\mu_{C2}(P_2)\Lambda \\ & \mu_{C3}(P_3)\Lambda\ldots,\mu_{CT-1}(P_{T-1})\Lambda\mu_{B_T}(S_T)\}\end{aligned} \tag{3.7}$$

This can be written as

$$\begin{aligned}& \mu_D(Pmax_1, Pmax_2, Pmax_3,\ldots,Pmax_{T-1}) \\ = {} & Max_{\{(P1,\,P2,\,P3,\,\ldots,\,PT-2)\varepsilon Y\}}\{Max_{\{(PT-1)\varepsilon Y\}}\{\mu_{C1}(P_1)\Lambda \\ & \mu_{C2}(P_2)\Lambda\mu_{C3}(P_3)\Lambda\ldots,\mu_{CT-1}(P_{T-1})\Lambda\mu_{B_T}(S_T)\}\} \\ = {} & Max_{\{(P1,P2,P3,\ldots,PT-2)\varepsilon Y\}}\{\{\mu_{C1}(P_1)\Lambda\mu_{C2}(P_2)\Lambda \\ & \mu_{C3}(P_3)\Lambda\ldots,\mu_{CT-2}(P_{T-2})\}\Lambda \\ & Max_{(PT-1\varepsilon Y)}\{\mu_{CT-1}(P_{T-1})\Lambda\mu_{B_T}(f(S_{T-1}, P_{T-1}))\}\}\end{aligned}$$

© 2001 by CRC Press LLC

where $\{\mu_{C1}(P_1)\Lambda\mu_{C2}(P_2)\Lambda\mu_{C3}(P_3)\Lambda\ldots,\mu_{CT-2}(P_{T-2})\}$ is not relevant to P_{T-1}, but $\{\mu_{CT-1}(P_{T-1})\Lambda\mu B_T(f(S_{T-1}, P_{T-1}))\}$ is.

If the fuzzy decisions satisfy the fuzzy goals in level $t = T - 1$, then the fuzzy membership function is $\mu_{B_{T-1}}(S_{T-1}) = Max_{(PT-1\varepsilon Y)}\{\mu_{C_{T-1}}(P_{T-1})\Lambda\mu_{B_T}(f(S_{T-1}, P_{T-1}))\}$, thus

$$\begin{aligned}&\mu_D(Pmax_1, Pmax_2, Pmax_3,\ldots,Pmax_{T-1})\\ &\quad = Max_{\{(P1,P2,P3,\ldots,PT-2)\varepsilon Y\}}\{\mu_{C1}(P_1)\Lambda\mu_{C2}(P_2)\Lambda\\ &\qquad \mu_{C3}(P_3)\Lambda\ldots,\mu_{CT-2}(P_{T-2})\Lambda\mu_{B_{T-1}}(S_{T-1})\}\end{aligned} \tag{3.8}$$

Similarly, if the fuzzy decisions that satisfy the fuzzy goals at level $t = T - 2$, then there will be

$$\begin{aligned}&\mu_D(Pmax_1, Pmax_2, Pmax_3,\ldots,Pmax_{T-1})\\ &\quad = Max_{\{(P1,P2,P3,\ldots,PT-3)\varepsilon Y\}}\{\mu_{C1}(P_1)\Lambda\mu_{C2}(P_2)\Lambda\\ &\qquad \mu_{C3}(P_3)\Lambda\ldots,\mu_{CT-3}(P_{T-3})\Lambda\mu_{B_{T-2}}(S_{T-2})\}\\ &\quad = \ldots = Max_{(P1\varepsilon Y)}\{\mu_{C1}(P_1)\Lambda\mu_{B2}(S_2)\}\\ &\quad = \mu_{B1}(S_1)\end{aligned} \tag{3.9}$$

In general if $\mu_{B_{T-1}}(S_{T-1})$, $\mu_{B_{T-2}}(S_{T-2}),\ldots,\mu_{B_1}(S_1)$ can be decided by following formulae

$$\mu_{B_{T-1}}(S_{T-i}) = Max_{(P_{T-1}\varepsilon Y)}\{\mu_{C_{T-i}}(P_{T-i})\Lambda\mu_{B_{T-i+1}}(S_{T-i+1})\} \tag{3.10}$$

$$S_{T-i+1} = f(S_{T-i}, P_{T-i}) \tag{3.11}$$

where $i = 1, 2, 3,\ldots, T$, then eventually the decision making will reach the initial level with

$$\mu_{B_1}(P_1) = \mu D(Pmax_1, Pmax_2, Pmax_3,\ldots,Pmax_{T-1})$$

The optimal manufacturing process given by formula (3.10) is denoted as $P_{(T-i)M}$

$$P_{(T-i)M} = d_{T-i}(S_{T-i}), (i = 1, 2, 3,\ldots, T)$$

where d_{T-i} is called decision function, it means that at a specified level ($t = T - i$), the best decision is to select the part state S_{T-i} and the manufacturing process $P_{(T-i)M}$.

Fuzzy Function, Fuzzy Goals, and Fuzzy Constraints

To use the above mathematic model for process planning, the fuzzy function introduced in expression (3.4), the fuzzy goals, and the fuzzy constraints need to be specified. To do this, following propositions are necessary.

First, manufacturing resources such as machines, tools, and fixtures used in a manufacturing system are already known and allow all manufacturing processes within the system to be defined, represented, and stored in a data base.

Second, an alternative plan is developed from an alternative manufacturing route which is formed due to alternative in-process features and alternative manufacturing processes.

Third, each manufacturing process corresponds only one transformed part state that can be predefined with the process in the data base.

© 2001 by CRC Press LLC

Fuzzy Function $S_{t+1} = f(S_t, P_t)$

The fuzzy function f can be regarded as a representation of such a statement as "if a part state S_t, then applying manufacturing process P_t to obtain the next part state S_{t+1}." It implies two relationships, one between S_t and P_t and the other between S_t and S_{t+1}.

Because of the fact that obtaining the geometric form of an in-process feature is the primary function of most manufacturing processes (which has been defined earlier as process function), the capability and the efficiency of a manufacturing process will not be considered in the fuzzy function. The fuzzy function can be established based only on the relationship between the geometric form and the process function. For machining process, the geometric form and the process function are represented by a motion pattern that is the combination of the machine tool movements and the cutting tool form used by the machining operation in the machining process [Zhao and Baines, 1992].

$$Motion\ Pattern\ =\ \sum_{k=1}^{n} M_k + T \tag{3.12}$$

where M_k and T are vectors representing a machine movement and the form of cutting tool (i.e., the cutting edge profile), respectively. k means the kth machine movement and n means that there are n machine movements in the pattern.

The relationship between S_t and P_t can then be described as follows

$$S_t\ ==\ Motion\ pattern\ ==\ P_t \tag{3.13}$$

where == stands for "equivalence to." Within the data base, machining process P_t is identified by its function which in turn is defined by the motion pattern of the machining operation. When the geometric form (also defined by a motion pattern) of the in-process feature in part state S_t is given, the machining process P_t can be automatically selected according to expression (3.13). In theory, the transformed part state of S_{t+1} can be computed by

$$S_{t+1}\ =\ S_t \cup Volume\ of\ removed\ material \tag{3.14}$$

where the volume of removed material can be calculated based on the motion pattern. According to the third proposition made earlier, when machining process P_t is selected, S_{t+1} can be automatically retrieved. Therefore geometry computation using expression (3.14) is not necessary in this case.

Fuzzy Goals G

A fuzzy decision B_T in set B should be an optimal manufacturing route. Such a route should satisfy fuzzy goals G in level $t = T$. G can be defined as the minimum manufacturing time, the minimum number of manufacturing processes (or set-ups), and/or the minimum dissimilarity among machines and tools.

Fuzzy Constraints C

If fuzzy decision C_T satisfies the set of fuzzy constraints C, then both C_T and B_T are subsets of set B. Therefore the ideal fuzzy decision D is

$$D\ =\ C_T \cap C_T$$

This means that the fuzzy goals and the fuzzy constraints have the same effect on multi-level decision making process.

© 2001 by CRC Press LLC

In process planning, a part state is specified mainly by the geometric form of in-process features, while a manufacturing process is primarily for creating the part state by obtaining the geometric form of the in-process features. This is the fact used earlier to establish the fuzzy function *f*. However, a manufacturing process cannot always be selected based on its function according to the geometric form of the in-process features. Due to the fact that each part state has specific technical requirements such as dimensions, tolerances and surface roughness, the feature position and orientation, and the feature relations (formed by physical interactions and dimensions and geometric tolerances), manufacturing processes must have the required capability to attain those requirements. This forms one type of constraints to the manufacturing process P_t when it is selected according to state S_t.

$$\textit{Process capability} > \textit{Technical requirements} \tag{3.15}$$

where the symbol $>$ means *higher than*. More specifically, for machining processes the above constraints can be defined with the technical requirements attainable by machining process P_t under specific machining details. The machining process is specified by a set of machining details which form the capability of the process. It is evaluated under the specified machining conditions by being given a membership value between 0 and 1.

Suppose machining processes $\{P_{t1}, P_{t2}, P_{t3},\ldots,P_{tc}\}$ $(c \leq M)$ all satisfy relation (3.13) regarding to a part state S_t, the most suitable process should be the one that has the maximum value of $\mu_{Ct}(P_{ti})$ $(i = 1, 2, 3,\ldots,c)$. $\mu_{Ct}(P_{ti})$ is determined by comparing the capabilities of processes $\{P_{t1}, P_{t2}, P_{t3},\ldots,P_{tc}\}$ with the technical requirements of the part at state S_t.

Since a machining process is considered as a time interval during which only one machining operation with one machine tool, cutting tool, and one set-up is involved, the capability of the process can be defined based on the attainable accuracy of that machine tool and cutting tool. It may result in process plans that are not the best, but it does not affect decision making.

Another type of constraint needs to be defined and is associated with process efficiency which is measured by the machining time of the machining process. Since machining time of a single process can be precisely calculated, it is not considered in $\mu_{Ct}(P_{ti})$, but it is used as the goal for achieving the minimum machining time of a machining route.

3.8 Process Planning with Multi-Level Fuzzy Decision Making

The above technique can be implemented as a process planning tool of multi-level fuzzy decision making. When decision making is started, all alternative processes in set Y for part state S_t are selected based on the part states and the in-process features by applying the rules described by expressions (3.1), (3.2), and (3.3). For machining processes, they can be selected according to relation (3.12), (3.13), (3.14), and (3.15). All of the processes and part states specified will conform to the fuzzy function $S_{t+1} = f(S_t, P_t)$ and produce a necessary table similar to Table 3.3 (see next section).

During decision making, the alternative in-process features of an individual part state need to be identified and the default transformed part state in each alternative manufacturing process must be defined. To start process planning with multi level decision making, part states $\mu_{B_T}(P_T)(P_T \varepsilon X)$ are evaluated and process constraints $\mu_{Ct}(P_{ti})$ $(t = 1, 2, 3,\ldots, T, P_{ti}\varepsilon Y, i = 1, 2,\ldots, M)$ are determined. The output will be the optimal process plans in terms of minimum manufacturing time, minimum number of operations, and minimum dissimilarity of machines and tools. Alternative process plans can also be generated when requested. Below described is a planning example based on metal cutting processes for machining the part as shown in Figure 3.1. The machining operations, machine tools, and cutting tools, used in the machining processes are chosen merely for demonstration purpose, the planning result may not be applicable for a specific machining application or machining system.

© 2001 by CRC Press LLC

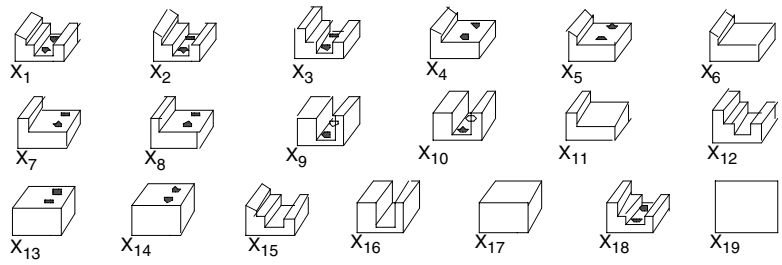


FIGURE 3.9 A fuzzy set of part states.

In the data base, six machining processes are stored and made available for machining the part. These processes form a set $Y = \{drilling, reaming, slotting, step\ milling, shaping, slab\ milling\}$. For this particular case, the number of manufacturing processes is six, i.e., $M = 6$.

The machine tools used in this case are all have the possibility of being selected for each of these processes. These machine tools are identified as: a drilling machine, a vertical milling machine, a horizontal milling machine, a shaper, and a lathe. Data about these machine tools with related cutting tools are stored in machine tool and cutting tool files and process capability files in the data base. Those files are accessed automatically through the index codes provided for each manufacturing process.

The part state model, as shown in Figure 3.6, is constructed using alternative in-process features and alternative machining processes (to save space, alternative machining processes between every two part states in each machining route are mostly indicated in Figure 3.6 by one single arrow line). The entire part state model is constructed with 18 different states, see Figure 3.9. Notice that X_4 and X_5 are different in that X_4 is the part state where the two holes are reamed and X_5 is the part state where two holes are drilled. An extra state X_{19} is added to the list, representing those that cannot be created by the machining processes in set Y. The total number of part states in set X is 19, i.e., $N = 19$.

As shown in Figure 3.6, the fuzzy decision making model contains 30 alternative machining routes and each route has six levels ($T = 6$) in total. The last level, which is the sixth level when $t = 6$, is represented as the leaf nodes of the part state tree. The decision making is started at this level with the evaluation of the possibility of every part state listed in Figure 3.9 to be created at this level. This valuation will generate the fuzzy membership of set X at the last level using the fuzzy function, $\mu_{B_6}(S_6)$. The result is given in Table 3.1.

Next the possibility of every manufacturing process in set Y to be used in each level is evaluated using $\mu_C(P_t)$. The valuation will produce the fuzzy membership constraints of each process at each level ($t = 1, 2, 3, 4, 5, 6$). The result is given in Table 3.2.

As shown in Figure 3.6, from each part state at one level, there are alternative part states transformed to the next level. In fact, each part state in set X could be possibly transformed by every manufacturing process in set Y into a different part state which is also in set X. For instance, the part state X_1 could be transformed by process Y_1 from X_{19} or by process Y_2 from X_2 or by process Y_3 from X_4 or by process Y_4 from X_{19} or by process Y_5 from X_3 or by process Y_6 from X_3. The entire alternative transformed part state can be practically created from the data base. The result is shown in Table 3.3.

Now according to expressions (3.10) and (3.11), the part states at the fifth level when $t = 5$ can be evaluated based on $\mu_{B_5}(S_5) = Max_{(P5 \varepsilon Y)}\{\mu_{C_5}(P_5) \Lambda \mu_{B_6}(f(S_5, P_5)\}$. For instance, let $S_5 = X_1$, according to Tables 3.1, 3.2, and 3.3, the fuzzy membership value of X_1 can be calculated as follows

$$
\begin{aligned}
\mu_{B5}(X_4) &= Max_{(i\,=\,1,2,3,4,5,6)}\{\mu_{C5}(Y_i) \Lambda \mu_{B6}(f(X_1, Y_i)\} \\
&= Max\{\mu_{C5}(Y_1) \Lambda \mu_{B6}(X_{19}),\ \mu_{C5}(Y_2) \Lambda \mu_{B6}(X_2),\ \mu_{C5}(Y_3) \Lambda \mu_{B6}(X_4), \\
&\quad \mu_{C5}(Y_4) \Lambda \mu_{B6}(X_{19}),\ \mu_{C5}(Y_5) \Lambda \mu_{B6}(X_3),\ \mu_{C5}(Y_5) \Lambda \mu_{B6}(X_3)\} \\
&= Max\{0.2 \Lambda 0, 0 \Lambda 0.1, 0.5 \Lambda 0.2, 0.9 \Lambda 0, 0.2 \Lambda 0.2, 0.3 \Lambda 0.2\} \\
&= Max\{0, 0, 0.2, 0, 0.2, 0.2\} = 0.2
\end{aligned}
$$

© 2001 by CRC Press LLC

TABLE 3.1 Valuation of Part States ($t = 6$)

X	$\mu_{B6}(S_6)$	X	$\mu_{B6}(S_6)$
X_1	0.0	X_{11}	0.9
X_2	0.1	X_{12}	0.4
X_3	0.2	X_{13}	0.5
X_4	0.1	X_{14}	0.7
X_5	0.2	X_{15}	0.2
X_6	0.3	X_{16}	0.8
X_7	0.2	X_{17}	1.0
X_8	0.6	X_{18}	0.1
X_9	0.3	X_{19}	0.0
X_{10}	0.6		

TABLE 3.2 Valuation of Constraints

Y	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6
$\mu_{C1}(P_1)$	0.0	0.8	0.7	0.4	0.3	0.6
$\mu_{C2}(P_2)$	0.4	0.8	0.8	0.4	0.2	0.6
$\mu_{C3}(P_3)$	0.8	0.4	0.7	0.7	0.3	0.5
$\mu_{C4}(P_4)$	0.3	0.2	0.6	0.9	0.1	0.0
$\mu_{C5}(P_5)$	0.2	0.0	0.5	0.9	0.2	0.3

TABLE 3.3 Alternative Transformed Part States Created by Processes in Y

X	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6
X_1	X_{19}	X_2	X_4	X_{19}	X_3	X_3
X_2	X_{15}	X_{19}	X_5	X_{19}	X_{18}	X_{18}
X_3	X_{19}	X_{18}	X_7	X_9	X_9	X_9
X_4	X_{19}	X_5	X_{19}	X_{19}	X_7	X_7
X_5	X_6	X_{19}	X_{19}	X_{19}	X_8	X_8
X_6	X_{19}	X_{19}	X_{19}	X_{19}	X_{11}	X_{11}
X_7	X_{19}	X_8	X_{19}	X_{13}	X_{19}	X_{19}
X_8	X_{11}	X_{19}	X_{19}	X_{14}	X_{19}	X_{19}
X_9	X_{19}	X_{10}	X_{13}	X_{19}	X_{19}	X_{19}
X_{10}	X_{16}	X_{19}	X_{14}	X_{19}	X_{19}	X_{19}
X_{11}	X_{19}	X_{19}	X_{19}	X_{17}	X_{17}	X_{17}
X_{12}	X_{19}	X_{19}	X_{11}	X_{16}	X_{11}	X_{11}
X_{13}	X_{19}	X_{14}	X_{19}	X_{19}	X_{19}	X_{19}
X_{14}	X_{17}	X_{19}	X_{19}	X_{19}	X_{19}	X_{19}
X_{15}	X_{19}	X_{19}	X_6	X_{19}	X_{12}	X_{12}
X_{16}	X_{19}	X_{19}	X_{17}	X_{19}	X_{17}	X_{17}
X_{17}	X_{19}	X_{19}	X_{19}	X_{19}	X_{19}	X_{19}
X_{18}	X_{12}	X_{19}	X_{14}	X_{10}	X_{10}	X_{10}
X_{19}	X_{19}	X_{19}	X_{19}	X_{19}	X_{19}	X_{19}

It can be seen from the above computation that the maximum value of $\mu_{B5}(X_1)$ at level five is achieved through $P_{max5} = Y_3$, Y_5, or Y_6, which means that if the part state is X_1 at level $t = 5$, the best decision made is "using process Y_3, Y_5 or Y_6." Thus there is a decision $d_5(X_1) = Y_3$, Y_5, or Y_6. Actually part state X_1 can hardly be machined either at this level or by process Y_3, Y_5, or Y_6, therefore the valuation of this decision is only 0.2 (1.0 is of course the ideal decision).

To find out the best decision at this level, let $S_5 = X_2, X_3, \ldots, X_{19}$, respectively, and carry on the similar computation for each part state. A complete valuation of the part states at level $t = 5$ can be obtained. Then let $S_4 = X_1, X_2, X_3, \ldots, X_{19}$, respectively, and performed the same computation, the valuation of the part states at level $t = 4$ can be obtained. By carrying on similar computation for S_3, S_2, and S_1, the part

© 2001 by CRC Press LLC

TABLE 3.4 Valuation of Part States ($t = 5, 4, 3, 2$, and 1)

X	$\mu_{B5}(S_5)$	$\mu_{B4}(S_4)$	$\mu_{B3}(S_3)$	$\mu_{B2}(S_2)$	$\mu_{B1}(S_1)$
X_1	0.2	0.2	0.5	0.5	0.6
X_2	0.2	0.3	0.5	0.6	0.0
X_3	0.3	0.5	0.4	0.6	0.0
X_4	0.2	0.2	0.3	0.3	0.0
X_5	0.3	0.3	0.3	0.0	0.0
X_6	0.3	0.1	0.0	0.0	0.0
X_7	0.5	0.2	0.3	0.0	0.0
X_8	0.7	0.3	0.0	0.0	0.0
X_9	0.5	0.2	0.2	0.0	0.0
X_{10}	0.5	0.2	0.0	0.0	0.0
X_{11}	0.9	0.0	0.0	0.0	0.0
X_{12}	0.8	0.6	0.0	0.0	0.0
X_{13}	0.0	0.2	0.0	0.0	0.0
X_{14}	0.2	0.0	0.0	0.0	0.0
X_{15}	0.3	0.3	0.5	0.0	0.0
X_{16}	0.5	0.0	0.0	0.0	0.0
X_{17}	0.0	0.0	0.0	0.0	0.0
X_{18}	0.6	0.5	0.6	0.0	0.0
X_{19}	0.0	0.0	0.0	0.0	0.0

TABLE 3.5 Values of Decision Functions $d_t(S_t)$ (* Means No Decisions)

X	$d_5(S_5)$	$d_4(S_4)$	$d_3(S_3)$	$d_2(S_2)$	$d_1(S_1)$
X_1	$Y_3Y_5Y_6$	Y_2Y_3	Y_6	Y_2	$Y_2Y_6Y_3Y_5$
X_2	Y_1Y_2	Y_1Y_3	Y_6	Y_6	*
X_3	Y_4Y_6	Y_3Y_4	Y_2	Y_2	*
X_4	Y_5Y_6	Y_2	Y_2	Y_2Y_6	*
X_5	Y_6	Y_1	Y_5Y_6	*	*
X_6	Y_6	Y_5	*	*	*
X_7	Y_4	Y_2	Y_2	*	*
X_8	Y_4	Y_1	*	*	*
X_9	Y_3	Y_2	Y_2Y_3	*	*
X_{10}	Y_3	Y_1	*	*	*
X_{11}	Y_4	*	*	*	*
X_{12}	Y_4	Y_3	*	*	*
X_{13}	*	Y_2	*	*	*
X_{14}	Y_1	*	*	*	*
X_{15}	Y_3Y_6	Y_3	Y_6	*	*
X_{16}	Y_3	*	*	*	*
X_{17}	*	*	*	*	*
X_{18}	Y_4	Y_4	Y_1	*	*
X_{19}	*	*	*	*	*

states at levels $t = 3, 2$, and 1 can be evaluated. The final results are listed in Tables 3.4 and 3.5. It can be seen from Table 3.5 that at time $t = 1$, there is only one part state $S_1 = X_1$. According to expression (3.9), it results in

$$\mu_D(Pmax_1, Pmax_2, Pmax_3, Pmax_4, Pmax_5) = \mu_{B1}(S_1) = 0.6$$

The machining process $Pmax_1$ can be Y_2, Y_3, Y_5, or Y_6. First considering $Pmax_1 = Y_2$, according to Table 3.3 the part state at level $t = 2$ can be obtained as

$$S_2 = f(X_1, Y_2) = X_2$$

© 2001 by CRC Press LLC

According to Table 3.5, $Pmax_2 = Y_6$. Similarly, at level $t = 3$, $S_3 = f(X_2, Y_6) = X_{18}$ and $Pmax_3 = Y_1$; at level $t = 4$, $S_4 = f(X_{18}, Y_1) = X_{12}$ and $Pmax_4 = Y_3$; at level $t = 5$, $S_5 = f(X_{12}, Y_3) = X_{11}$ and $Pmax_5 = Y_4$. Finally at level $t = 6$, $S_6 = f(X_{11}, Y_4) = X_{17}$. It can be seen from Table 3.1 that $\mu_{B6}(X_{17}) = 1.0$. Therefore the decision is

$$(Pmax_1, Pmax_2, Pmax_3, Pmax_4, Pmax_5) = (Y_2, Y_6, Y_1, Y_3, Y_4)$$

This decision can be evaluated based on expression (3.6),

$$\begin{aligned}\mu_D(Y_2, Y_6, Y_1, Y_3, Y_4) &= \mu_{C1}(Y_2)\Lambda\mu_{C2}(Y_6)\Lambda\mu_{C3}(Y_1)\Lambda \\ &\quad \mu_{C4}(Y_3)\Lambda\mu_{C5}(Y_4)\Lambda\mu_{B6}(X_{17}) \\ &= 0.8\ \Lambda\ 0.6\ \Lambda\ 0.8\ \Lambda\ 0.6\ \Lambda\ 0.9\ \Lambda\ 1.0 = 0.6\end{aligned}$$

Then considering $Pmax_1 = Y_3, Y_5$, and Y_6, respectively, the other decisions can be made by following the same computation process. The results are as follows

$$\begin{aligned}(Pmax_1, Pmax_2, Pmax_3, Pmax_4, Pmax_5) &= (Y_3, Y_2 \text{ or } Y_6, Y_5 \text{ or } Y_6 \text{ or } Y_2, Y_1, Y_4) \\ (Pmax_1, Pmax_2, Pmax_3, Pmax_4, Pmax_5) &= (Y_5, Y_2, Y_1, Y_3, Y_4) \\ (Pmax_1, Pmax_2, Pmax_3, Pmax_4, Pmax_5) &= (Y_6, Y_2, Y_1, Y_3, Y_4)\end{aligned}$$

The valuation of the above decisions are

$$\begin{aligned}\mu_D(Y_3, Y_2 \text{ or } Y_6, Y_5 \text{ or } Y_6 \text{ or } Y_2, Y_1, Y_4) &= 0.3 \text{ or } 0.4 \\ \mu_D(Y_5, Y_2, Y_1, Y_3, Y_4) &= 0.3 \\ \mu_D(Y_6, Y_2, Y_1, Y_3, Y_4) &= 0.6\end{aligned}$$

From the above analysis, the optimal machining routes are $(Y_2, Y_6, Y_1, Y_3, Y_4)$ and $(Y_6, Y_2, Y_1, Y_3, Y_4)$. This means that there are two optimal decisions which are "using machining route starting from reaming, through slab milling, drilling, and slotting to step milling" and "using machining route starting from slab milling through reaming, drilling, and slotting to step milling." The two decisions are listed as follows.

Solution 1 is found by fuzzy set method to be as follows:

(node) FINISHED PART
    by (link) REAMING HOLES ON MACHINE M3
        with (attribute as) 54 to
(node) HOLES TO REAM (S13)
    by (link) MILLING ON MACHINE M1
        with (attribute as) 10 to
(node) ANGLE FACE TO CUT (S131)
    by (link) DRILLING HOLES ON MACHINE M4
        with (attribute as) 22 to
(node) HOLES TO DRILL (S1313)
    by (link) SLOTTING ON MACHINE M2
        with (attribute as) 50 to
(node) SLOT TO CUT (S13132)
    by (link) MILLING STEP ON MACHINE M2
        with (attribute as) 32 to
until finally reaches (node) RAW MATERIAL

© 2001 by CRC Press LLC

Solution 2 is found by fuzzy set method to be as follows:

(node) FINISHED PART
by (link) MILLING ON MACHINE M1
with (attribute as) 10 to
(node) ANGLE FACE TO CUT (S11)
by (link) REAMING HOLES ON MACHINE M3
with (attribute as) 54 to
(node) HOLES TO REAM (S112)
by (link) DRILLING HOLES ON MACHINE M4
with (attribute as) 22 to
(node) HOLES TO DRILL (S1122)
by (link) SLOTTING ON MACHINE M2
with (attribute as) 50 to
(node) SLOT TO CUT (S11222)
by (link) MILLING STEP ON MACHINE M2
with (attribute as) 32 to
until finally reaches (node) RAW MATERIAL

3.9 Conclusions

Process planning is application oriented and company specific. The vagueness of knowledge, the uncertainty of decision making, the time-dependence of planning, and the alternatives of problem-solving in decision making interrelate with each other and pose a great challenge to conventional process planning approaches. If these problems as described in this chapter could be resolved, CAPP would become a reasonable mature subject within the field of CIM.

Fuzzy set theory combined with AI techniques has formed powerful tools in dealing with the similar problems in other fields. Interestingly, the decision making in process planning involves a large proportion of multi-level problems with fuzzy logic and fuzzy knowledge and AI techniques have been used as one of the most common methods of problem-solving in process planning, but the application of multi-level decision making and the fuzzy set theory in process planning seems overlooked. The work presented above provides encouraging evidence to show that multi-level decision making with both AI techniques and fuzzy set theory can provide effective solutions to the highlighted problems. It is hoped that this will bring the attention of researchers of the same interests to explore the potential of the techniques in the areas.

This chapter also emphasised the requirement that a practical CAPP system should be flexible and user-friendly. Being flexible is twofold, capable of providing alternative process plans for similar manufacturing requirements and being easily adopted to different manufacturing environments. Being user-friendly largely means that users do not have to be a process planning expert. Some traditional process planning systems with built-in logic and rules offer little flexibility; others with generic definition of rules and logic require too much human intervention. The techniques presented in the text could provide a better solution. For instance, alternative decisions are included, the best decision can be made when all the alternatives are evaluated according to certain criteria under specific conditions. Considering alternative decisions will gain the flexibility and valuation of those alternatives will enable users to apply some predefined constraints (e.g., design requirements, resources, and machine specifications) without using specialised process planning knowledge. The definition of constraints and criteria (called fuzzy constraints and fuzzy goals in the paper) is crucial to the resultant process plans, but is irrelevant to the way of decision making. Users can specify the criteria and constraints, according to their own applications, without changing the system, thus the system is able to be adopted in different manufacturing environments.

The part state tree and the fuzzy state model provides an effective searching space not only for multi-level decision making, but for process optimisation and process plan selection. Every plan can be considered

© 2001 by CRC Press LLC

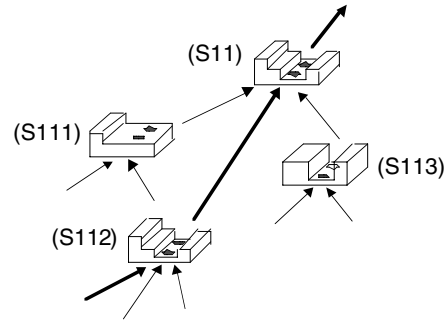


FIGURE 3.10 Eliminating part states by in-process feature elements.

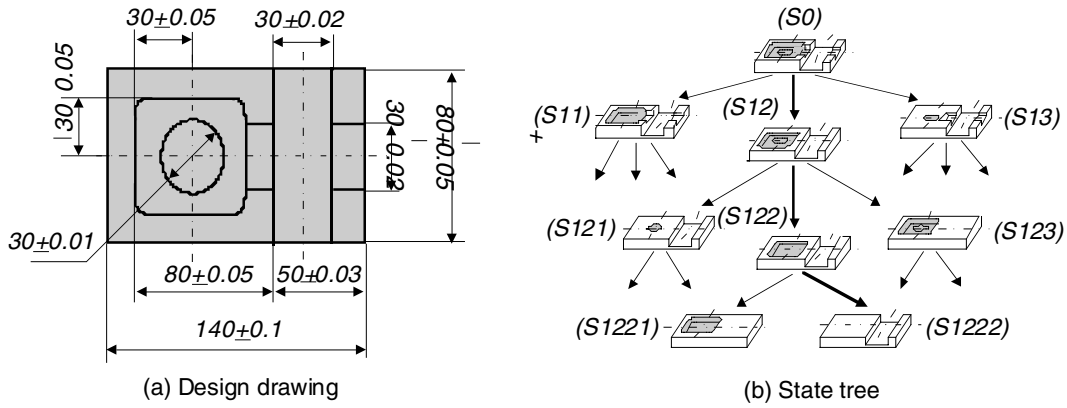


FIGURE 3.11 Master feature.

as a variable described by process parameters such as process function, process capability, and process efficiency. In reality, however, building a complete process planning solution domain is time consuming and can become a explosive problem. To reduce the dimensions of the state tree by controlling the number of the alternatives, there are two options. One is that computer programmes can be designed to let users assist computers to form the state tree interactively and selectively. The tree contains only the states that users consider useful for his or hers applications. In this way, the software is simply a data collection programme that entirely relies on the user to make decisions in reducing the number of part states. The other option is to provide the software with a feature-processing data base. The programme selects part states and the required processes from the data base. The number of states is reduced by considering the arrangement of in-process feature elements in each process. This can be explained by the part states shown in Figure 3.10.

The part in the current state S_{11} could be machined from three states, S_{111}, S_{112}, and S_{113}. If, for instance, the positions of the holes relative to both the slot and the step are controlled by tight tolerances, the slot and the step must be previously cut and used as reference elements (datum) during the reaming process. In this case, only the state S_{112} should be chosen, the other two states do not need to be included.

The size of the solution domain can be further reduced by identifying the master feature of individual part states that could dominate the processes that follow. Machining a specific feature in one state may require other features in previous states to be machined first. This specific feature will affect the following processes. As depicted by the example shown in Figure 3.11, the upper slot (viewed horizontally in the design drawing) of the part shown in Figure 3.11(a), for example, is such a master feature because the machining of the slot requires the square pocket to be machined first in order to create a space so that the cutting tool can cut through the entire slot.

© 2001 by CRC Press LLC

After the pocket is machined, the hole has to be drilled without resetting the workpiece. This will help to attain the position (controlled by two dimensions, both being 30 $\pm$ 0.05 mm) of the hole relative to the pocket. The lower slot (viewed vertically in the design drawing) is to be machined before the pocket to be used as a reference element during machining the hole and the other features. Therefore, the lower slot is the master feature for obtaining the dimension 80 $\pm$ 0.05 mm during machining the pocket. In this case, the number of states will be reduced into one state path S_0-S_{12}-S_{122}-S_{1222}. Obviously, specifying master features in a part state tree can reduce the dimensions of the process plan domain. In the state tree, a state to which a master feature belongs is in fact the root of a substate tree. If the domain contains several master features, the state tree can be entirely a combination of those substate trees and contains minimum number of alternative state paths.

Every manufacturing process is fully specified by its constituent in the part state tree and is selected according its function, capability, and efficiency. Since all possible processes are to be evaluated based on these three factors, decision making in a dilemmatic situation does not have to be made. For example, there exist two processes, *A* and *B*, to machine an in-process feature. If process *A* is used, the feature is to be machined with higher cost due to the use of special tools and fixtures. If process *B* is used, the feature is to be machined with lower cost, but longer set-up time. On one hand, *B* should be selected since it costs less. On the other hand, *A* should also be selected since its set-up time is shorter. In this situation, the decision is hard to make. For this case, the part state decision making domain is intended to include all the choices and leaves the decision to be made later when all decisions are evaluated.

Further work should focus on the effective methods for constructing the part state tree. The bottleneck is the acquisition and representation of process planning knowledge from design and manufacturing. An immediate task is to provide CAD modelling functions and tools and make them available for process planning functions to manipulate in-process features and part state transformation. Also it must be realised that the future success of process planning will not be a truly automated CAPP system, but a CAPP framework that provide users with sufficient flexibility and necessary computing support. A clear line should be drawn between CAPP system designers and CAPP system users to divide the process planning functionality within the framework into those that the system designers should provide and those that the users should add on. Such a clarification will make a step forward toward the standardisation of process plans and the common agreement on process planning functions.

References

Alting, L. and Zhang, H. C., 1989, Computer aided process planning: the state-of-art survey, *Int. J. of Prod. Res.*, 27 (4), 553–585.

Bhaskaran, K., 1990, Process plan selection, *Int. J. Prod. Res.*, 28 (8), 1527–1539.

Bryan, A. C. and Steven, R. R., 1991, ALPS: a language for process specification, *Int. J. Comput. Integrated Manuf.*, 4 (2), 105–113.

Case, K. and Gao, J., 1993, Feature technology: an overview, *Int. J. Comput. Integrated Manuf.*, 6 (1 & 2), 2–12.

Chang, T. C., 1990, *Expert Process Planning for Manufacturing,* Addison-Wesley, Reading, MA.

Chang, T. C. and Wysk, R. A., 1985, *An Introduction to Automated Process Planning Systems*, Prentice-Hall, Englewood Cliffs, NJ.

Curtis, M. A., 1988, *Process Planning*, John Wiley & Sons, New York.

Gulesin, M. and Jones, R. M., 1994, Face oriented neighbouring graph (FONG): a part representing scheme for process planning, *Comput. Integrated Manuf. Syst.*, 7 (3), 213–218.

Gupta, T., 1990, An expert system approach in process planning: current development and its future, *Comput. and Industrial Eng.*, 18 (1), 69–80.

ISO 10303-1, 1992, STEP Part 1: overview and fundamental principles, ISO TC184/SC4/WGPMAG Document N43, NIST, Gaithersburg, MD.

Jasthi, S. R. K, Rao, P. N., and Tewari, N. K., 1995, Studies on process plan representation in CAPP systems, *Comput. Integrated Manuf. Syst.*, 8 (3), 173–184.

Klir, G. J. and Folger, T. A., 1988, *Fuzzy Sets, Uncertainty and Information*, Prentice-Hall, Englewood Cliffs, NJ.

© 2001 by CRC Press LLC

Kusiak, A., 1991, Process planning: a knowledge-based and optimisation perspective, *IEEE Trans. Robotics and Autom.*, 7 (3), 257–266.

Kusiak, A. and Finke, G., 1988, Selection of process plans in automated manufacturing systems, *IEEE J. Robotics and Autom.*, 7 (4), 397–402.

Larsen, N. E., 1991, Effects from use of alternative routings, The 24th CIRP International Seminar on Manufacturing Systems, (June), Denmark.

Larsen, N. E., 1993, Methods for integration of process planning and production planning, *Int. J. Comput. Integrated Manuf.*, 6, (1 & 2), 152–162.

Lee, S., Wysk, R. A., and Smith, J. S., 1995, Process planning interface for a shop floor control architecture for computer-integrated manufacturing, *Int. J. Prod. Res.*, 33 (9), 2415–2435.

Maropoulos, P. G., 1995, Review of research in tooling technology, process modelling and process planning (part II: process planning), *Comput. Integrated Manuf. Syst.*, 8 (1), 13–20.

Mill, F. G., Salmon, J. C., and Fedley, A. G., 1993, Representation problems in feature based approaches to design and process planning, *Int. J. Comput. Integrated Manuf.*, 6 (1 & 2), 27–33.

Mizumoto, M., Fukami, S., and Tanaka, K., 1979, Some methods of fuzzy reasoning, advances in fuzzy set theory and applications, M. M. Gupta, R. K. Ragade, and R. R. Yager, Eds., 117–136, North-Holland, Amsterdam.

Niebel, B. W., 1965, Machinised process selection for planning new design, ASME Paper No. 737.

Pratt, M. J., 1993, Applications of feature recognition in the product life-cycle, *Int. J. Comput. Integrated Manuf.*, 6 (1 & 2), 13–19.

Schenk, D. E., 1966, Feasibility of automated process planning, Ph.D. thesis, Purdue University, West Lafayette, IN.

Singh, N. and Mohanty, B. K., 1991, A fuzzy approach to multi-objective routing problems with application to process planning in manufacturing system, *Int. J. Prod. Res.*, 29 (6), 1161–1170.

Zadeh, L. A., 1965, Fuzzy sets, *Information and Control*, No. 8, 338–353.

Zadeh, L. A., 1991, Fuzzy logic, principles, applications and perspective, public lecture, April 18, University of Oklahoma, Norman, OK.

Zhang, H. C. and Alting, L., 1993, Computerised manufacturing process planning systems, Chapman & Hall, London.

Zhang, H. C. and Huang, S. H., 1994, A fuzzy approach to process plan selection, *Int. J. Prod. Res.*, 32 (6), 1265–1279.

Zhang, H. C. and Huang, S. H., 1995, Applications of neural networks in manufacturing: a state-of-the-art survey, *Int. J. Prod. Res.*, 33 (3), 705–728.

Zhao, Z. X., 1992, Generative process planning by conjugative coding design and manufacturing information, Ph.D. thesis, Staffordshire University, Stafford, UK.

Zhao, Z. X., 1995, Process planning with multi-level fuzzy decision-making, *J. Comput. Integrated Manuf.*, 8 (4), 245–254.

Zhao, Z. X., 1997, A methodology management approach to computerised process planning, *Int. J. Comput. Integrated Manuf.*, 10 (1–4), 83–91.

Zhao, Z. X. and Baines, R. W., 1992, Conjugative coding design and manufacturing information in computer integrated manufacturing system, *Int. J. Prod. Res.*, 30 (10), 2313–2333.

Zhao, Z. X. and Baines, R. W., 1994, Software design for sequencing machining operations, Advances in Manufacturing Technology VIII, Proceedings of Tenth National Conference on Manufacturing Research. September 5–7, 309–313, Loughborough, UK.

Zhao, Z. X. and Baines, R. W., 1996a, A CADCAM framework attributed to methodology management technology (MMT), Proceedings of Advanced Manufacturing Processes, Systems, and Technology (AMPST 96), March 26–27, 725–734, Bradford University.

Zhao, Z. X. and Baines, R. W., 1996b, The impact of process plan standardisation on process planning functions and system, *Adv. Manuf. Technol.*, 10, 76–80.

Zhao, Z. X., Baines, R. W., and Blount, G. N., 1993, Definition of generic relationships between design and manufacturing information for generative process planning, *J. Comput. Integrated Manuf. Sys.*, 6 (3), 204–212.

© 2001 by CRC Press LLC

4

A Computer-Aided and Integrated Machining System

4.1 Design Philosophy
4.2 User Interface
4.3 Feature Extraction
4.4 The Feature Recognizer
AutoCad Data • Processor • Analyzer • Initial Feature Builder • Machining Feature Builder
4.5 The Knowledge Base
4.6 Factual Knowledge
Data Content
4.7 Procedural Knowledge
Selection of Operations • Sequencing of Operations • Selection of Tools • Selection of Machining Conditions • Generation of Tool Paths
4.8 Concluding Remarks

Swee-Hock Yeo
Nanyang Technological University

In order to arrive at a suitable solution, be it for process planning or part programming, the pressing issues faced in an integrated manufacturing system are in the following.

1. Interpretation of CAD data for machining-based operations.
2. System integration in the activities involved in process planning, machinability data selection, and tool path generation.
3. Update information arising from progress in manufacturing technology.
4. Decision making relating to manufacturing tasks.
5. Optimization strategy suitable for CNC processing capabilities.

The concept of an integrated machining system for the generation of production plans is introduced in this work. The development of methodologies to address the above issues is presented. This chapter expounds on the work reported by Yeo, Rahman, and Wong [1991] and Yeo [1995].

4.1 Design Philosophy

In this section, the concept of an integrated knowledge-based machining system is described. A software tool, GOLDWORKS™ [Gold Hill, 1989] with COMMON LISP as the programming platform, is used for the development work. The software tool requires Microsoft Windows® in a PC which can be feasibly deployed in a wide range of industries. The system is aimed to provide a vital step to a totally integrated

© 2001 by CRC Press LLC

manufacturing environment. The integrated machining system (IMS) begins with the interpretation of a part drawing and transformation of the geometrical data into a frame structure of machining features. The system then assigns appropriate operations to all features identified. Automatic tool selection of holders and inserts follows before planning the entire operation sequence. For each operation, the machining conditions are determined. Finally, tool paths are generated for a selected CNC lathe.

The assumption made in developing the proposed IMS is that component set-ups with suitable work-holding parameters are provided so that the solution generated by the IMS can be implemented. Proper set-up involves spatial reasoning, and it is a difficult task that an experienced machinist learns during an apprenticeship. Varying methods of solution exist from simple rules-of-thumb, such as threshold ratio between length and diameter and analytical methods such as that of explained by Hinduja and Huang [1989]. The problem of specifying the appropriate set-up is further complicated by the variety of work-holding devices which include face plates and fixtures, mandrels, jaw-type chucks, step chucks, collets, and magnetic and vacuum chucks.

The architecture of the system that has been developed in this research is illustrated in Figure 4.1. The system is comprised of a user-interface for interaction, a feature recognizer, and a knowledge base that contains facts and rules. Figure 4.2 shows the various types of approaches/strategies that have been used for the machining planning tasks. It also outlines the sections of this chapter for each of these tasks.

The inferential strategy used for the decision-making process of the IMS is based on the goal directed forward chaining technique (which integrates both forward and backward chaining techniques) [Yeo, 1995].

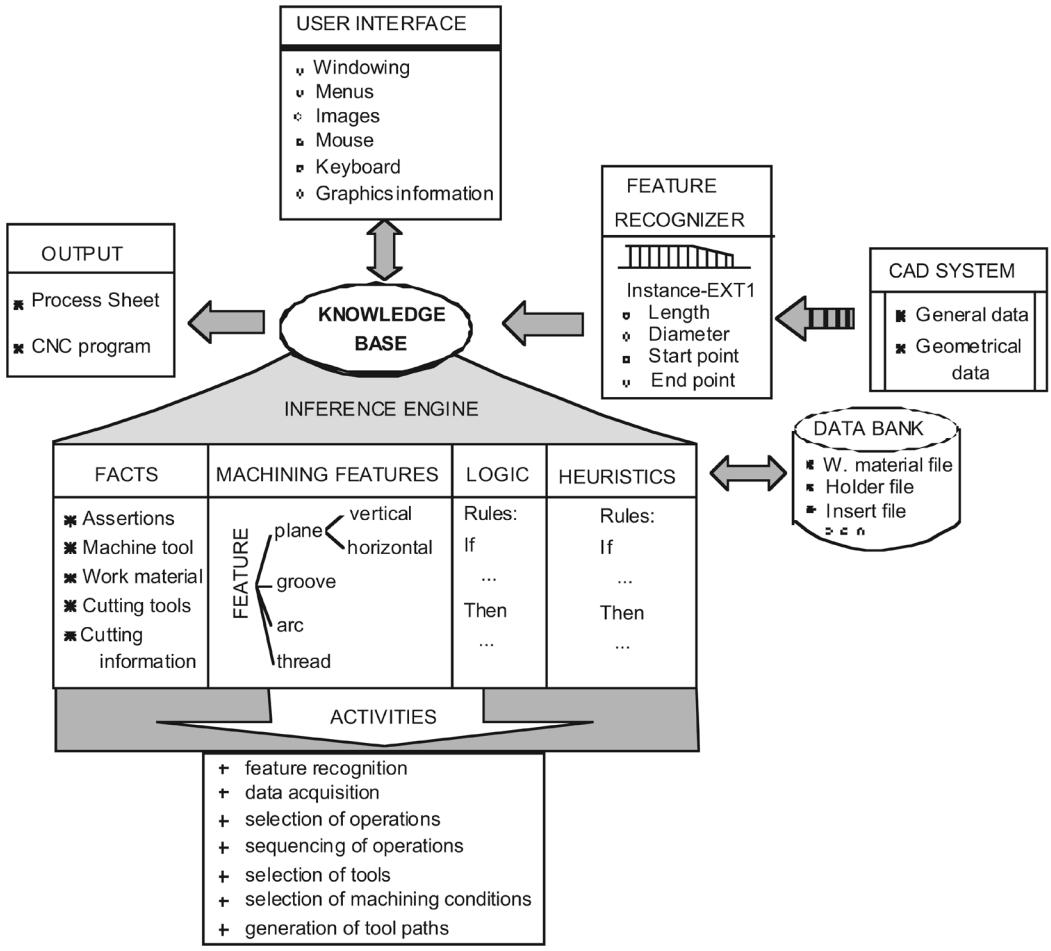


FIGURE 4.1 Architecture of the integrated machining system.

© 2001 by CRC Press LLC

Machining planning tasks	Strategies/Solution Approach		
	Numerical calculations	Logical determination	Creative decisions/ heuristics
(a) Feature recognition	✔	✔	✔
(b) Selection of operations	✔	✔	
(c) Sequencing of operations		✔	✔
(d) Selection of tools	✔	✔	✔
(e) Selection of cutting conditions	✔		✔
(f) Generation of tool paths	✔		

FIGURE 4.2 Decisions modes in the integrated machining system.

For a given goal, the chaining method uses rule sets to cluster forward-rules together, and each rule set is activated when an enabling pattern is matched and queried. The control structure consists of eight stages where each stage is made up to a rule set as depicted in the following with a brief description in the right column.

```
(DEFINE-RULE SELECTION-CONTROL                  ; production rule with a series of enabling
                                                  patterns
(RUN-STATE FEATURE-RECOGNITION DONE)            ; define machinable features, stock size, etc.
(RUN-STATE INTERNAL-OPERATION DONE)             ; selection of internal operations
(RUN-STATE EXTERNAL-OPERATION DONE)             ; selection of external operations
(RUN-STATE OPERATION-SEQUENCING DONE)           ; assignment of operation sequence
(RUN-STATE TOOLHOLDER-SELECTION DONE)           ; selection of toolholders
(RUN-STATE INSERT-SELECTION DONE)               ; selection of inserts
(RUN-STATE MACHINING-CONDITION DONE)            ; selection of machining conditions
(RUN-STATE TOOL-PATH-GENERATION DONE)           ; generation of tool paths
THEN
(RUN-STATE SELECTION COMPLETED)                 ; query of the attempt to fire the rule
```

The inferencing mechanism is fired by backward chaining via the THEN part of the control structure with the pattern named RUN-STATE SELECTION COMPLETED. Each rule set is activated sequentially when its enabling pattern (e.g., RUN-STATE FEATURE-RECOGNITION DONE) is matched and the inference mechanism performs forward chaining of all the rules related to the rule set. The benefits of having such methods are having one group of rules fired before another group and improving code efficiency of the matching process in which only appropriate rules for a particular state are used.

Outputs obtained from the IMS are composed of two main parts namely (see Figure 4.1).

1. A process sheet for each set-up, which is a set of instructions consisting of the following.
   - Enumerated operations with lists of machining features.
   - Cutting tools (holders, inserts, drills, etc.) for the operations planned.
   - Positions of cutting tool mounted on the turret.
2. A CNC part program consisting of machine codes.

© 2001 by CRC Press LLC

4.2 User Interface

The purpose of this user interface is to provide interaction between the user and the system, data input, and data modification during the machining planning process. The user interface is driven by means of a series of menu-driven facilities and graphic images. These include the following.

1. A screen layout with a row of commands for pull-down menus.
2. A window with a set of buttons for performing machining planning tasks.
3. A window for schematic display of a task.
4. A status bar window to monitor the progress of the machining planning tasks.
5. A window for the textual explanation of the problem solving procedures.
6. Pop-up menus for the data input.

In addition, there is a help facility for obtaining information regarding tooling, work materials, etc. Error trap facilities are also incorporated in the system, for example, input of a null value or an erroneous value.

4.3 Feature Extraction

Since this research deals with 2-D operations for producing rotationally symmetric parts on lathes, a plan view of the upper half cross-section of the part drawing of the workpiece will suffice. The form of the profile is restricted to straight lines and arcs. The feature recognizer requires access to part descriptions stored in a CAD database. The approach that has been adopted in this work is to develop a module that involves inductive extraction of the geometrical data of a part created in a commercial CAD system. Its DXF (in AutoCAD®) facility provides the output format required by the IMS. Before any operation planning activity can take place, a set of features must be formulated for abstraction. Since this work is directed at metal removal processes, machinable features are used. A machinable feature is any geometric surface or combination of geometric surfaces which form a volume that can be shaped by cutting tool operations. The term "form features" is sometimes used.

The necessary information about each feature is organized by means of a frame representation. This knowledge representation provides modularity and convenient accessibility of the knowledge base.

4.4 The Feature Recognizer

The feature recognizer module has five steps which are shown in Figure 4.3 as described:

AutoCad Data

A section view of the top half of a part is created using AutoCAD®. Line-types of each entity and their significance have been defined as shown in Figure 4.4 so that the CAD data can be used effectively in the next step. All the entities representing the model and the stock size are exported using the AutoCad DXF format.

Processor

The Processor accepts a DXF data file and processes it into a form acceptable to the analyzer. The COMMON LISP list-data structure has been used to store the DXF data into a temporary file. An example of the temporary file is as follows:

```
(0 SECTION 2 ENTITIES
 0 POINT 8....
 0 LINE 8....
 0 ARC 8....
 ....
 0 ENDSEC 0 EOF)
```

© 2001 by CRC Press LLC

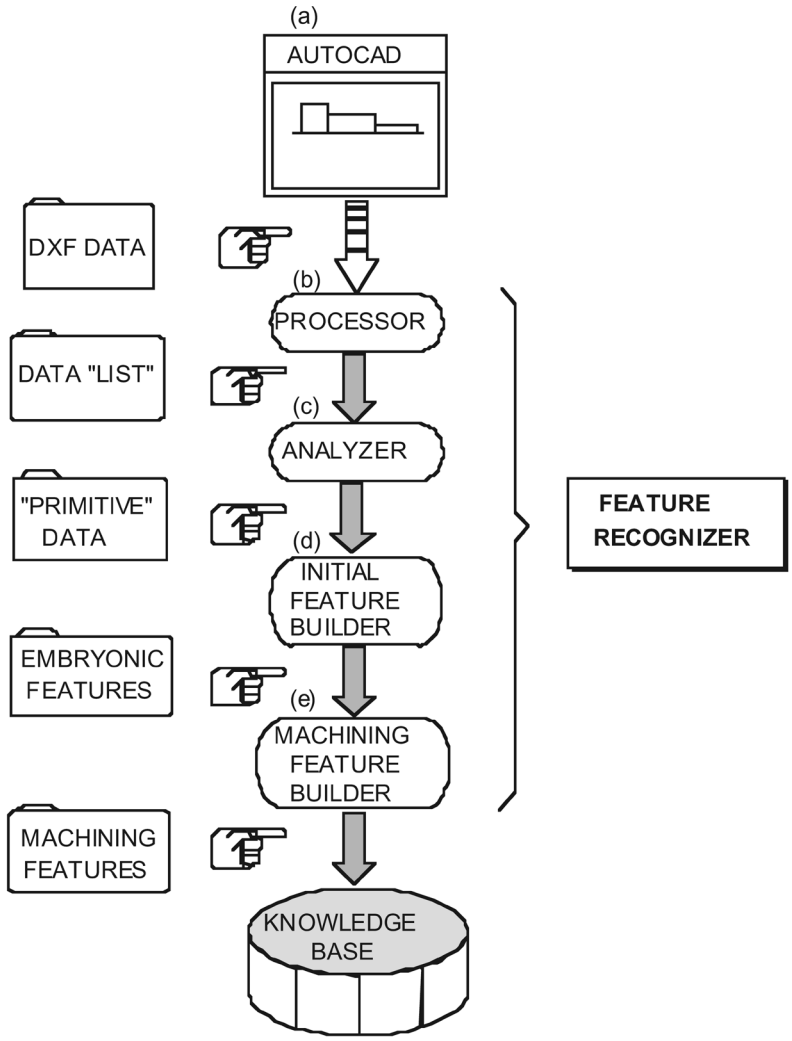


FIGURE 4.3 Structure of the feature recognizer.

The temporary list file is then processed further by removing integer notations and group entities into a more refined list data file as follows:

((LINE (178.2 142.2) (221.3 165.2)) .... (ARC (141.7 151.7) 5.0 (180.0 270.0)))	; sub list for each ; entity type

Each entity type and geometrical data is contained in a sub list.

Analyzer

The analyzer determines the nature of each entity and assigns to each entity a primitive feature which defines functionality. The primitive features include vertical_line, horizontal_line, slant_line, threaded_line, convex_up, convex_down, concave_up, and concave_down. The various line types are

© 2001 by CRC Press LLC

DESCRIPTION	LINE TYPES	USE
Continuous	———	Visible or external surface
Hidden	– – – – – – – –	Internal surface
Dashed	— — — — —	External threaded surface
Phantom	——— – – ———	Internal threaded surface
Border	— — · — — ·	Stock size of material

FIGURE 4.4 Significance of the line type.

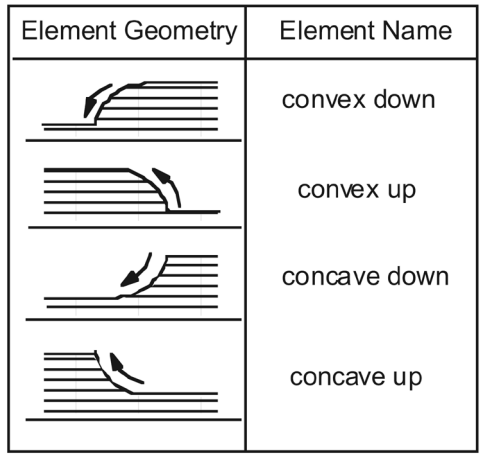


FIGURE 4.5 Element library of arc types.

easily understood. The various arc types are illustrated in Figure 4.5. Their significance is used for tool selection. For the example used in ("convex up"), the sub list resulting from this analysis gives the primitive list data as follows:

```
((LINE (178.2 142.2) (221.3 165.2) (PRIMITIVE SLANT_LINE))
    ....
(ARC  (141.7 151.7) 5.0 (180.0 270.0) (PRIMITIVE CONVEX_UP))
)
```

Each sub list describes the types of primitive. The entire primitive list data is sorted in order, starting from the left-most primitive of a part.

Initial Feature Builder

Using a frame lattice structure shown in Figure 4.6, the initial feature builder converts the primitive list data into embryonic objects (i.e., instances of frames) which are exclusively geometric. The machining features of a component are represented in the knowledge base in the form of frames with hierarchical

© 2001 by CRC Press LLC

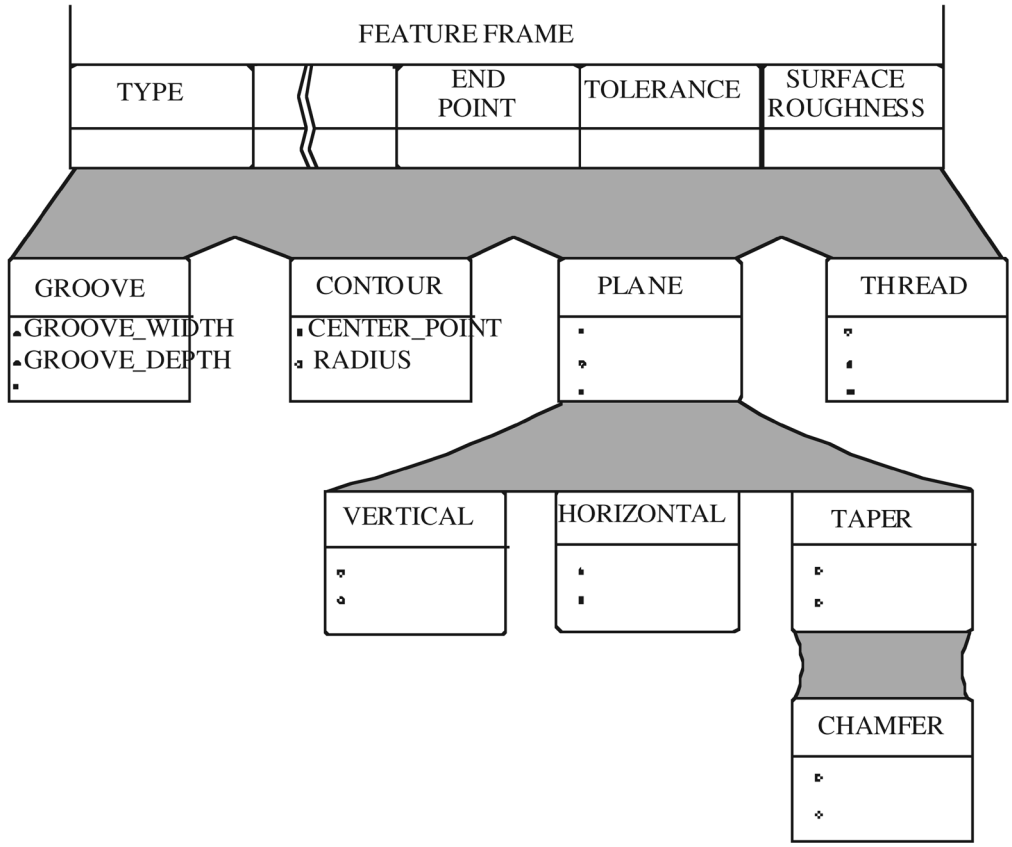


FIGURE 4.6 Frame lattice structure of the machining features for a rotationally symmetric part.

relationships. The lattice structure consists of eight types of frame used for a rotationally symmetric part, which are: plane, horizontal, vertical, taper, arc, chamfer, groove, and thread. Seven of these types of frame may have associated instances. Two slots in the feature frame are used to provide connectivity to adjacent features. The plane-type frame groups three other frames which are all concerned with straight lines.

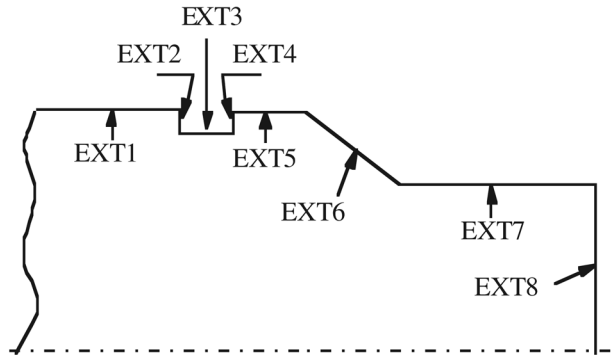
An example of a taper feature instance in the language syntax form is shown in Figure 4.7. The EXT6 instance is related to the taper frame which is a child frame of the plane frame. Beside the representation of feature types, each instance is linked to its adjacent features (i.e., EXT5 and EXT7), thus forming a linked list. For convenient identification, each instance is uniquely named and enumerated, for example EXT and INT imply external and internal feature-types respectively, and the numbers appended to them are arranged in order starting from the left-most feature of the part.

Machining Feature Builder

The machining feature builder makes a complete set of all the machinable features of a part for the knowledge base. Production rules are used in the pattern matching process. The rules include recognition of grooves, chamfers, and threads. Surface roughness and geometric tolerances can be added manually to the attributes of a feature as appropriate.

While the recognition of chamfers and threads are easily implemented, the recognition of grooves requires a more detailed geometrical treatment. A groove (or recess) in a machining profile is generally

© 2001 by CRC Press LLC




```
Define-instance EXT6
    (:is taper)
    (angle_ratio -0.13)
    (length 16.12)
    (type external)
    (location 62.0)
    (diameter 44.0)
    (start_point (70.0 20.0))
    (end_point (54.0 22.0)
    (fwd_adj_surface EXT7)
    (bwd_adj_surface EXT5)
    (surface_roughness)
    (tolerance)
    (status)
    (operations)
```

FIGURE 4.7 An example of a taper feature instance.

described as being confined by two adjacent boundaries. This definition is very broad and might lead to inefficient automatic tool selection. Grooves may be required in a wide variety of forms such as: circlip grooving, O-ring grooving, face grooving, deep grooving, wide grooving, and undercutting.

In this work, the recognition of groove is tool-oriented. There are various possibilities for machining a groove feature consisting of a horizontal element bounded by two vertical elements as illustrated in Figure 4.8. The use of two tools, that is, two longitudinal tools (Figure 4.8a) or a longitudinal tool and a grooving tool (Figure 4.8b), requires two tool positions on the turret; thus there is an economic implication. One grooving tool (Figure 4.8c) could be used to solve the problem of limited turret capacity. It may not be economical if a large number of passes is required, i.e., for a wide groove.

To produce a rectangular groove feature, i.e., both the vertical elements parallel to the *X*-axis and the bottom of the groove parallel to the *Z*-axis, the external tool must have an approach angle of more than 90°, and the largest possible trailing edge angle. An insert shape of 35° included angle would be suitable. Figure 4.9 shows the best choice among the standard tools available for external operations. Using a 25 mm square shank toolholder, the groove depth (l) should not be more than 42 mm and width (*w*) at least 64 mm, depending on the clearance provided, hence, the critical *w/l* ratio is equal to 1.524.

© 2001 by CRC Press LLC

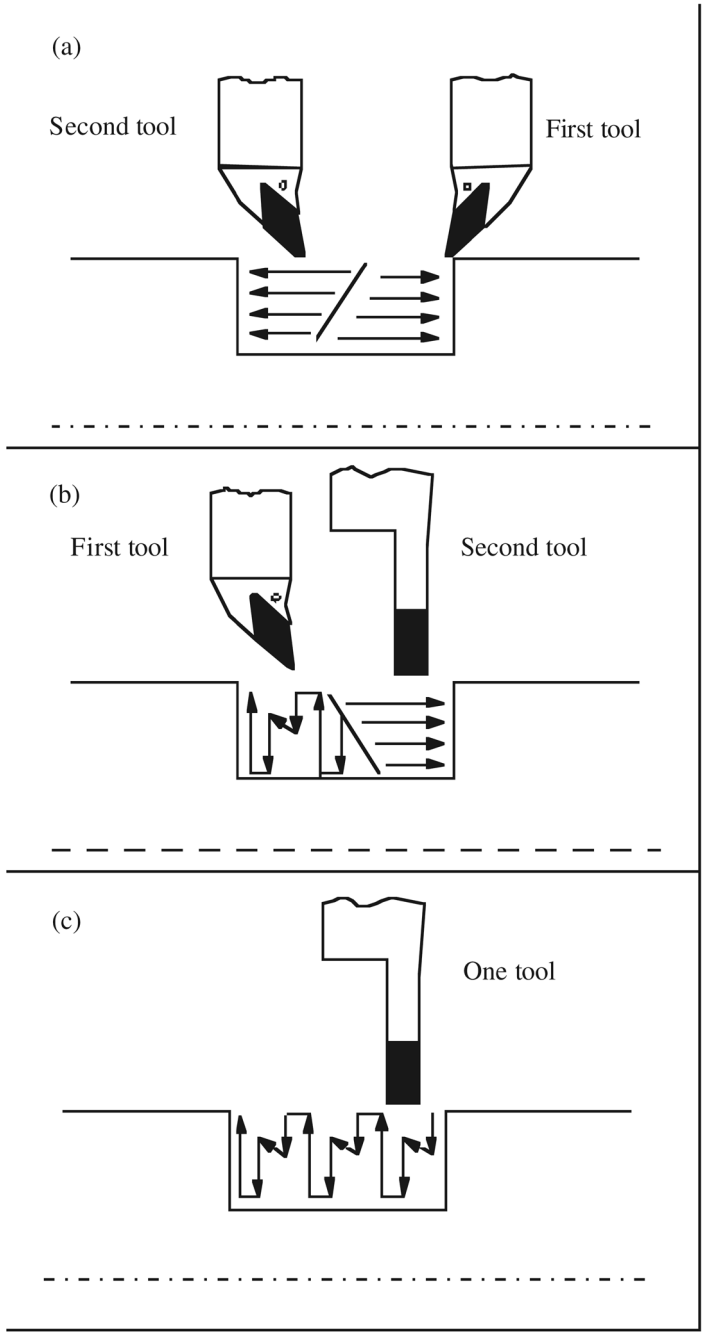


FIGURE 4.8 Three possibilities for machining a groove feature.

As compared with this tool-type, a standard 25 mm square shank grooving tool has a *w/l* ratio of 1.63 (see Figure 4.10). Though the *w/l* ratio for grooving tools is marginally less than that of the external tool, the size of the groove can be as deep as 16 mm with 26.2 mm width. The former tool cannot produce this shape.

© 2001 by CRC Press LLC

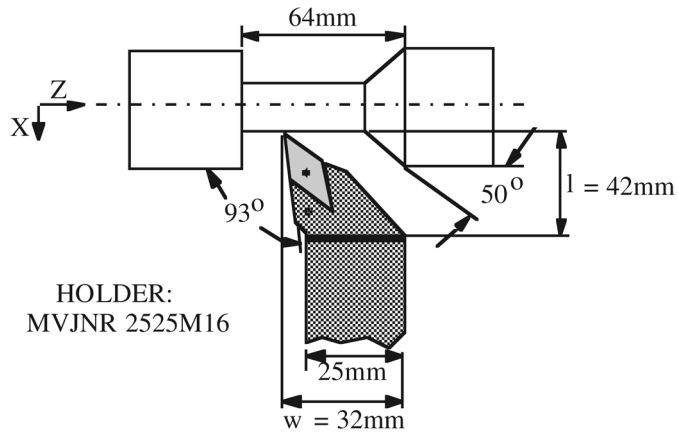


FIGURE 4.9 Geometrical constraint of an external cutting tool to machining a recess.

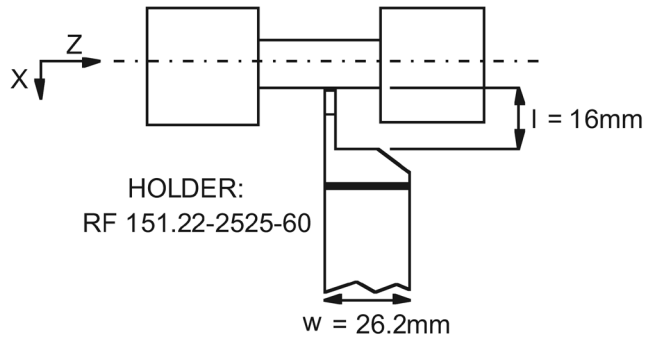


FIGURE 4.10 External grooving tool.

The machining feature builder is designed in such a manner that modification to the knowledge base can be done easily. Thus, the production rule for a groove can be extended to provide for a wide groove, a narrow groove, and a deep groove. Other form tools for cutting specific grooves or recesses are not considered in the present work. An example of a rule to recognize a groove-feature is as follows:

IF
surface, F1 is adjacent to horizontal surface, F2, and
horizontal surface F2 is adjacent to surface, F3,
F1 & F3 are 90° to F2, and
F2 with width of less than 26.2 mm
and
F1 with depth of less than 16 mm
THEN
surfaces F1, F2, F3 form a feature GROOVE

Using the example given in Figure 4.7, Features EXT2, EXT3, and EXT4 satisfy the groove-feature rule and are combined into a groove feature; thus, a frame-based approach to formulate a lattice structure of machining features in a rotationally symmetric part has provided an efficient means to represent a generic model. The organization of the lattice structure is suitable for efficient pattern matching of rules. The methodology has been used for a wide range of machining features in a part, and provides the flexibility to change feature definitions with minimal effort.

© 2001 by CRC Press LLC

4.5 The Knowledge Base

The types of knowledge acquired are a combination of factual (or declarative) knowledge and procedural knowledge, which involves selection of: operations, operation sequencing, selection of tools, selection of machining conditions, and generation of tool paths.

4.6 Factual Knowledge

A sizable amount of data is required for machining decision making. Specifications of work materials, toolholders, inserts, etc., must be stored in the files of the data base management system, with maintenance of these files being carried out as necessary.

As indicated in Figure 4.11 an interface between the knowledge base and the data base files must be employed for the retrieval of relevant data. External data base files, where data searching must be done sequentially on the index key until the search key field is matched, must be accessed by the interfacewthis way, besides improving the computational time, the data base management system can also be used to serve other organizational functions, such as tool management which is important to improve overall productivity. The crucial issues, namely system integration, ease of data accessibility and system flexibility are thus taken into account.

Data Content

The data base files for recommending machining data required by the knowledge-based system have been described by Yeo, Wong, and Rahman [1991]. In order to search and retrieve data from the work material data base files, the dbase-action frame structure shown in Figure 4.12 serves as an interface to the external files.

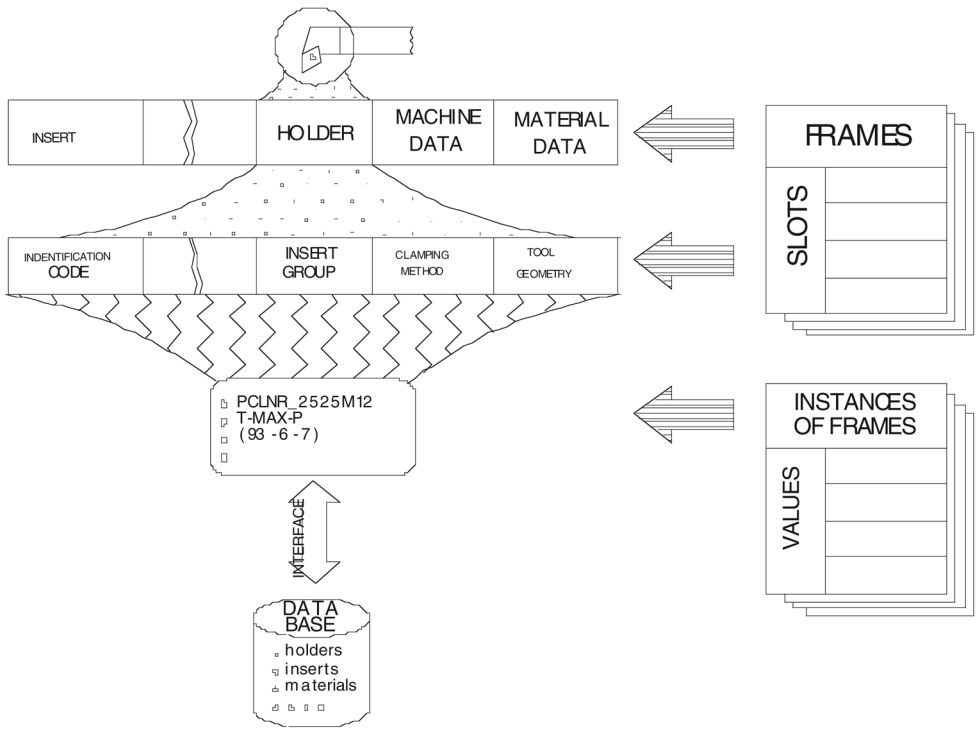


FIGURE 4.11 Data acquisition system for the knowledge base.

© 2001 by CRC Press LLC

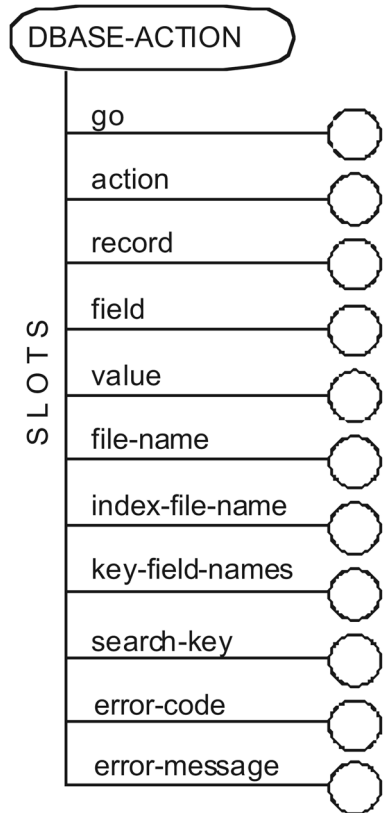


FIGURE 4.12 Frame structure of a dbase-action.

For example, a data base file is opened by using the function:

```
(Defun open dbase (interface)
      (setf (slot value interface 'action) :open file)
      (setf (slot value interface 'go) :yes))
```

The variable interface is an instance of dbase-action frame. The action slot-name of ':open-file' simply opens a data base file when the go slot-name is ':yes.' A further action named ':find-record' searches the file until the search-key slot-name is matched and the result (i.e., record) is asserted in the value slot-name.

The part description of an instance named THE-PART-INFO is shown in Figure 4.13. The assertion of the stock size is obtained from the DXF file. The material specifications are retrieved and asserted to the relevant slots using the above interface procedure. External data accessing is done quickly with ease even with a large data file size. If numerous data queries are made on the same file, for example the cutting tool data files used for checking tool availability and tool selection, it would be convenient to convert the data into a frame system. A particular machine tool with its attributes is represented as an instance of the machine tool frame THE-MACHINE-INFO in Figure 4.14.

Data base structure for cutting tools can amount to more than 20,000 combinations of tool modules [Evershiem, Jacobs, and Wienand, 1987]. It is important to provide efficiently in the data base structure for the following.

1. Ease of maintenance.
2. Complete and concise attributes of tools.
3. Linkages between holders, inserts, etc.

A tool database structure consisting of 21 data fields has been formalized and is used for external and internal holders as well as for drills. Examples of a tool file containing three types of tool are shown in Table 4.1.

© 2001 by CRC Press LLC

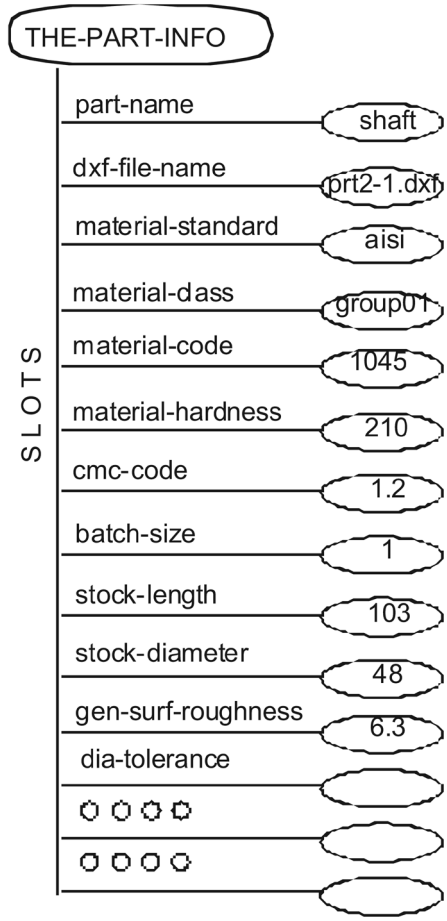


FIGURE 4.13 Part description of an instance named THE-PART-INFO.

The purpose of each data field is as follows.

1. Identification code. This field is used to designate a unique identification for a tool holder or a drill.
2. Insert clamping method. This field is used for four types of clamping systems which are designated in the ISO standard, i.e., C, M, P, and S types. The C type provides a positive rake which is usually preferred for finishing operations, while the M type provides rigidity and permits use of double sided inserts. The P type offers unobstructed chip flow in addition to the advantages for M type. The S type is used for finishing operations and provides a larger variety of insert shapes. The C type is largely used in ceramic inserts.
3. Coupling (shank) size. This field is used for the specification of shank height, shank width, and tool length, which affect the stiffness of the tool. Its size is restricted by the turret tooling capability.
4. Hole diameter. This field is used to ensure that a boring bar is able to function without colliding into the workpiece. For example, a 25 mm bar diameter should have a minimum hole diameter of 32 mm when the bar is placed at the centre of rotation [Sandvik Coromant, 1990]. In the case of drilling, the data field refers to the drill diameter.
5. Tool function. This field is used for the classification of operations (sometimes referred to as process classification). The hand of tool (i.e., left, right, or neutral) specified in the ISO standard is insufficient to describe the full capability of a tool such as that illustrated in Figure 4.15. The right-hand tool, PDJNR, shown in Figure 4.15a is used for profiling while PCLNR, shown in Figure 4.15b is used for longitudinal turning and out-facing, as well as for in-facing.

© 2001 by CRC Press LLC

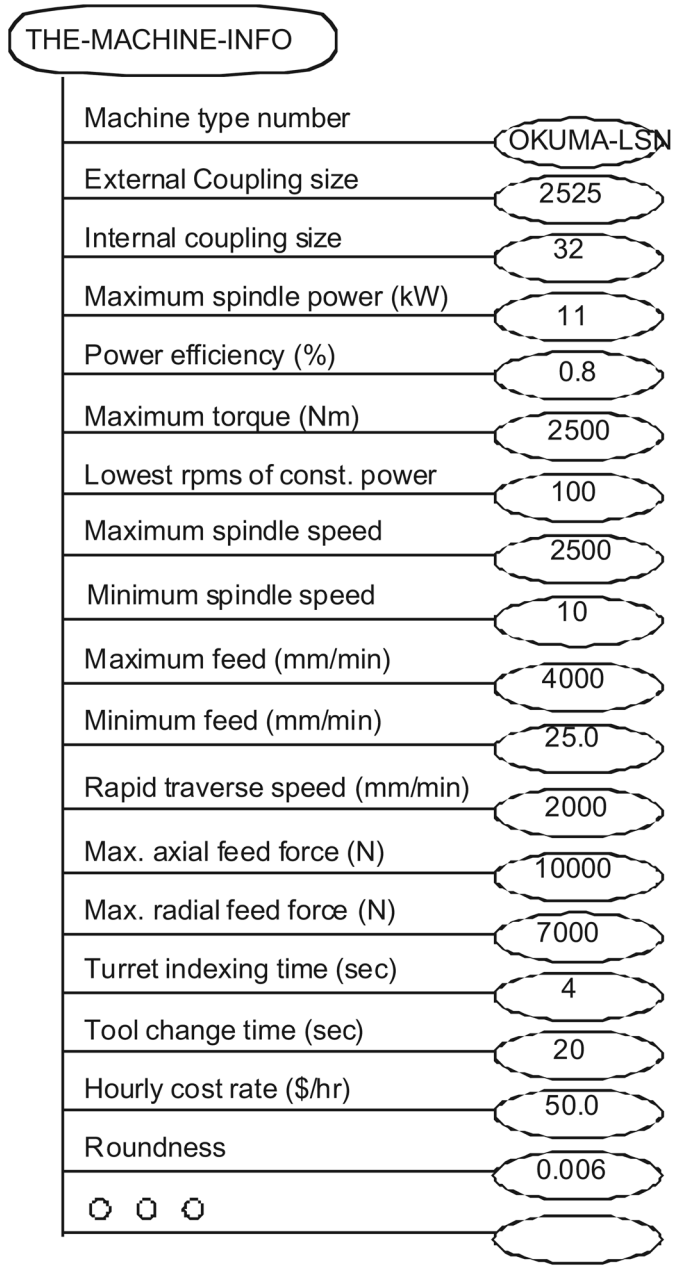


FIGURE 4.14 Description of the Okuma LSN Lathe.

6. Entering angle. This field is used as part of the tool specification defined in ISO 3002/1-1977. The effects of this geometry characteristic are well established in machining processes. For example, a smaller entering angle increases the ability of a tool to take a larger feed. On the other hand, a larger entering angle is able to reduce tool chatter and vibration.
7. Trailing angle. This field is used to check for tool collision, particularly important in an in-copying situation.
8. Inclination angle. This field is used as part of the tool specification in which its primarily influence is the chip flow direction.

© 2001 by CRC Press LLC

TABLE 4.1 Examples of the Tool File Structure Used to Describe a Turning Tool, a Boring Tool, and a Drill

No.	Description	External Holders	Internal Holders	Drill
1	Identification code	PCLNR_2525M_12	S20S-SDUCR_11-M	R416.1-0180-20-05
2	Insert clamping method	T-MAX P	T-MAX U	T-MAX U
3	Coupling (shank) size	2525	20	18
4	Hole diameter (mm)	nil	25	nil
5	Tool function	(face-in-turn axial-turn face-out-turn)	(profile-bore)	(drilling)
6	Entering angle (deg)	95	93	nil
7	Trailing angle (deg)	nil	30	nil
8	Inclination angle (deg)	−6	nil	9
9	Normal rake angle (deg)	−6	−6	nil
10	Insert group	C12P	D11U	L03U
11	Cost	55	55	50
12	Adaptor	nil	nil	R146.1-801-6
13	Factors	nil	nil	nil
14	Groove range (mm)	nil	nil	nil
15	*Z/X* dimensions (mm)	(32 150)	(250 13)	nil
16	Maximum cutting length (mm) (or drill cut. length; groove cut depth)	nil	nil	55
17	Holder life (no. of edges/holder life)	400	400	nil
18	Number of inserts	1	1	2
19	Drill tip angle (deg)	nil	nil	114
20	Material grade	nil	nil	nil
21	Hand	right	right	right

9. Normal rake angle. This field is used as part of the tool specification which influences the power consumed in a machining process.
10. Insert group. This field is used to group inserts which are applicable to particular types of tool holder, i.e., for inserts that fall in a group named C12P, a tool holder 'PCLNR 2525 M12' can be used.
11. Cost. This field specifies the cost of the holder.
12. Adaptor. This field specifies appropriate adaptors to be used in conjunction with holders such as boring bars and drills.
13. Factors. This field specifies merits of three factors influencing the tool selection process, namely: entering angle, insert clamping system, and shank size. A weightage which is dependent on a cutting situation is applied to the merits of a tool to determine its suitability, i.e., in a roughing operation, insert clamping system of P-type is most suitable.
14. Groove range. This field gives a range of groove-width which is dependent on the capability of its holder.
15. Z/X dimensions. This field gives the *Z*- and *X*-dimensions relating to the overall tool size.
16. Maximum cutting length. This field is used for drilling and grooving tools. For drilling tools, this refers to the cutting length of its body, while for grooving, it implies the groove cut-depth as shown in Figure 4.10.
17. Holder life (no. of edges/holder life). This field refers to the number of cutting edges a toolholder can be used during its life-span. 400 edges per toolholder's life is normally used.
18. Number of inserts. This field specifies the number of inserts used in a tool holder. By default, one insert is used in a toolholder. In the case of an insert-type drill, two inserts may be used.
19. Drill tip angle. This field specifies the drill tip angle, i.e., a 114° tip angle is used for U-drills.
20. Material grade. This field specifies the material grade for solid cutting tools such as delta drills.
21. Hand. This field is used to identify right hand, left hand, or neutral tools.

© 2001 by CRC Press LLC

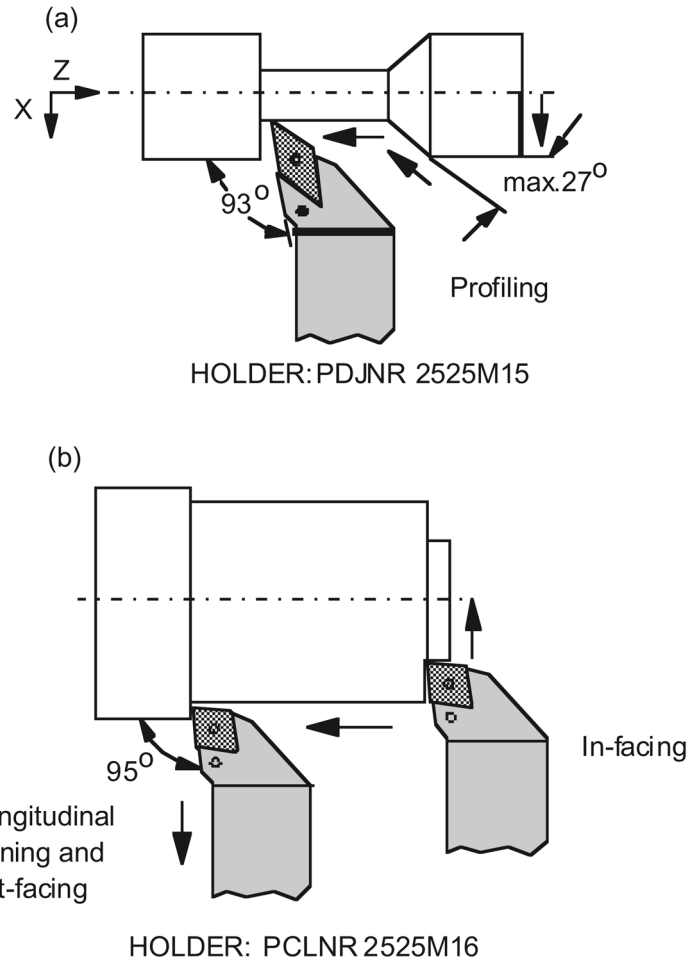


FIGURE 4.15 Some examples of tool functions for generating features.

Similarly, information about inserts is stored in an insert file database structure where each record consists of nine data fields. An example of an insert file containing three types of insert is shown in Table 4.2. The purpose of each data field is as follows:

1. Identification code. This field is used to identify a unique insert type.
2. Insert group. This field is used to categorize inserts into various groups, such as T12P. For example, an insert identity of TNMG_16_04_08-MF_GC415 is classified as insert group named T12P.
3. Insert shape types. This field is used to describe insert shapes according to letter symbols (as in ISO 1832: 1985). For example, round inserts, square inserts, and triangular inserts have symbols of R, S, and T, respectively.
4. Nose radius/pitch/width. This field is used to specify a nose radius, pitch, or width of a tool tip. The tool tip may be for turning, threading, or grooving.
5. Grade. This field gives the nominal material grade specified by the tool manufacturer, such as GC415.
6. ISO application area. This field provides the equivalent carbide grade according to ISO 513:1975.
7. Chipbreaker. This field refers to the chip breaker type used in the insert.
8. Number of cutting edges per insert. This field specifies the number of edges in an insert.

© 2001 by CRC Press LLC

TABLE 4.2 Example of Various Insert Types

No.	Description	Turning Inserts	Threading Inserts	Grooving Inserts
1	Identification code	TNMG_16_04_08-MF_GC415	R166.0G-16MM01-100_GC225	R154.91-3_300 S1P
2	Insert group	T16P	166UL	154-5
3	Insert shape types	CN	Y	Z
4	Nose radius/pitch/width	0.8	1.00	3.00
5	Grade (tool material)	GC415	GC225	S1P
6	ISO application area	P15	P25	P10
7	Chipbreaker	MF	nil	nil
8	No of cutting edges/insert	6	3	2
9	Max. cutting edge length (usable)	2.50	nil	4
10	Cost	8	8	7

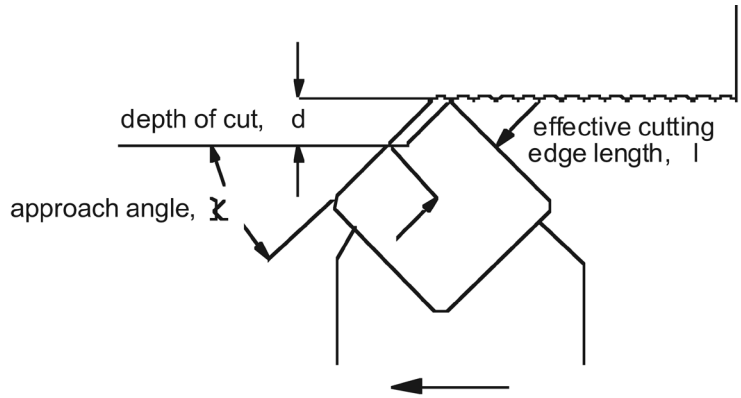


FIGURE 4.16 Dependency of cutting edge length on the entering angle and depth of cut.

9. Maximum cutting edge length. This field specifies the maximum allowable cutting edge length, l_{max}, of an insert to be able to cope with a particular depth of cut. The maximum value is based on the recommendation provided in the tool manufacturer's literature. The value given will then be used to compare with the actual cutting edge length of the cutting process as shown in Figure 4.16, requiring a condition of $l_{max} \geq$ l. For example, if l_{max} is 2.5 mm, a tool of 60° entering angle is used to cut at 2 mm depth of cut, then:

$$l = \frac{\mathrm{d}}{\sin \kappa}$$

$$l = \frac{2}{\sin \kappa} = 2.3 \text{ mm} \leq l_{max} \text{ (i.e., the tool is suitable)}$$

10. Cost. This field specifies the cost of the insert.

The data-types in the file system discussed above provide the tooling factual knowledge of the knowledge base.

© 2001 by CRC Press LLC

4.7 Procedural Knowledge

The procedural knowledge of the knowledge base consists of (a) selection of operations, (b) operation sequencing, (c) selection of tools, (d) selection of machining conditions, and (e) generation of tool paths.

Selection of Operations

A manufacturing operation may be defined as an elementary process using one tool, one fixturing of the part, and operation on a contiguous volume of material, [Weill, Spur, and Eversheim, 1982; and Zdeblick, 1983]. For example, cylindrical turning in several cuts with one type of tool will eventually reach the desired state of the component in a single manufacturing operation.

Much effort has been expended to develop an efficient strategy for this task. In the independent feature approach as shown in Figure 4.17, the planning process is repeated for each individual operation selected for each feature which is treated independently of the others. The result of each planning process is an operation plan. For the example given in Figure 4.7, the operation plan for each feature is as depicted in Figure 4.18. The major weakness of this strategy is that feature relations are not considered; the tools used for each feature may require numerous tool changes on the machine tool and the individual cutter paths may have to be modified before they are merged.

Instead of selecting the operations separately for each feature, a production rule may be used to determine which features can be produced by a particular operation. An example of a rule for a turning operation in the language syntax form is as follows:

```
IF
 (instance ?feature is horizontal_surface      ; if there is a horizontal feature
           with type external                  ; of external type
           with-unknown roughing? ?)           ; but no roughing operation is assigned to it
THEN
 (evaluate (make-axial-turn-operation          ; then assign relevant operation to the feature
           ?feature))
```

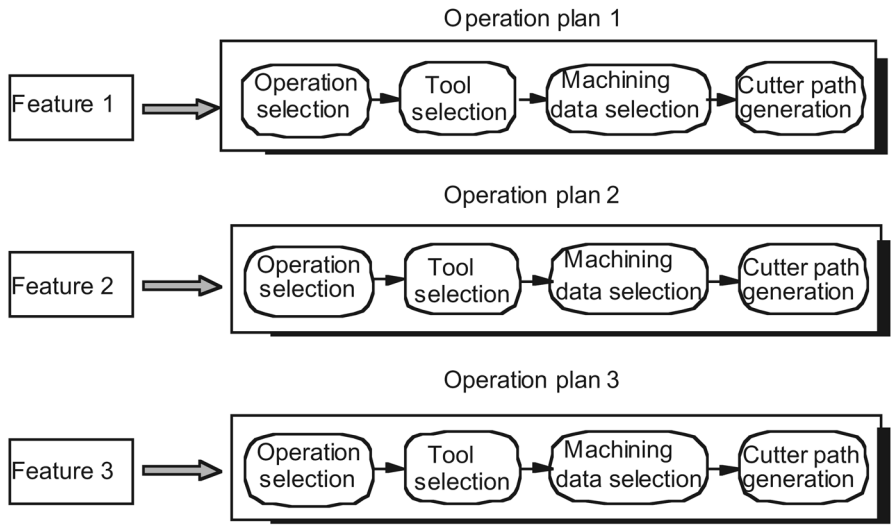


FIGURE 4.17 Planning based on independent feature approach.

© 2001 by CRC Press LLC

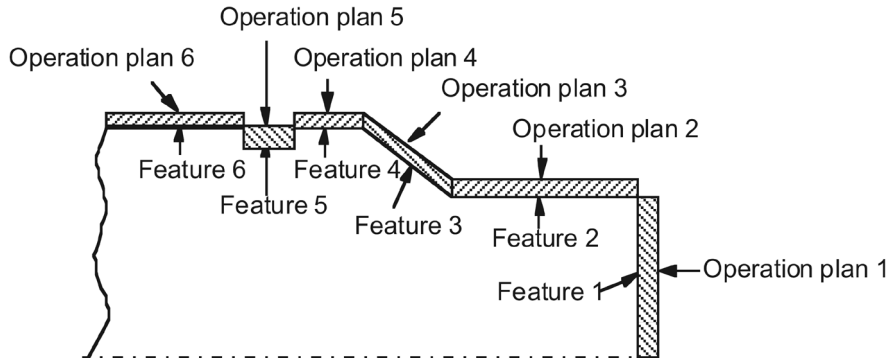


FIGURE 4.18 Operation planning for each feature.

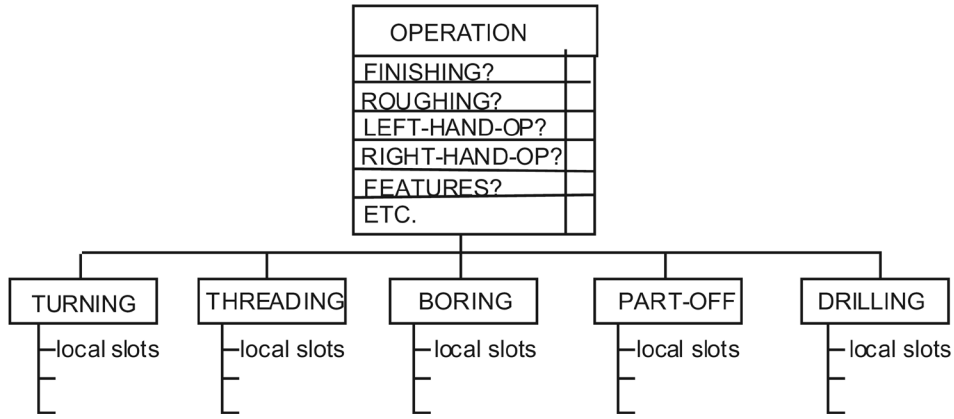


FIGURE 4.19 Planning based on operation capability.

There are two aspects to the selection of operation, namely (a) types of operation, and (b) types of feature. Variables are designated by a name preceded with "?". The variable '?FEATURE' refers to an instance of the frame-type representing the feature being planned and it requires two conditions within the antecedent part of the rule. The consequent 'THEN' part consists of a LISP function which assigns an appropriate operation according to the feature-type. Thus, by formulating the structure of the various types of operation shown in Figure 4.19, features that require a particular operation can be grouped together. This approach was explored during the preliminary stage of the research but was found to be inefficient because of its rigid structure. The problems in this strategy was similar to that of the independent feature approach because the selection of operation-types involved the selection of the hand of the tool and thus implicitly involved tool selection as well.

The method finally adopted is to use the relationships of all the features in a part. This approach to operation selection has been successfully demonstrated for hole-making operations [Yeo, Rahman, and Wong, 1990] and is conceptually represented in Figure 4.20. Appropriate operations are assigned to the 'OPERATIONS' slot of each feature (e.g., see Figure 4.7) and thereafter all the features with similar operations are grouped and sorted for operation sequencing.

The operation selection procedures are formulated in a rule set in which drilling, boring, threading, turning, and grooving are the basic operations that have been included. The type of machining must also be specified: rough cut, to remove the bulk of the stock; finish cut, to perform a final cut to meet the desired quality surface after the rough cut; and semi-finish cut, when less accuracy is acceptable than with a finish cut. Figure 4.21 shows the various operation-types used for operation selection.

© 2001 by CRC Press LLC

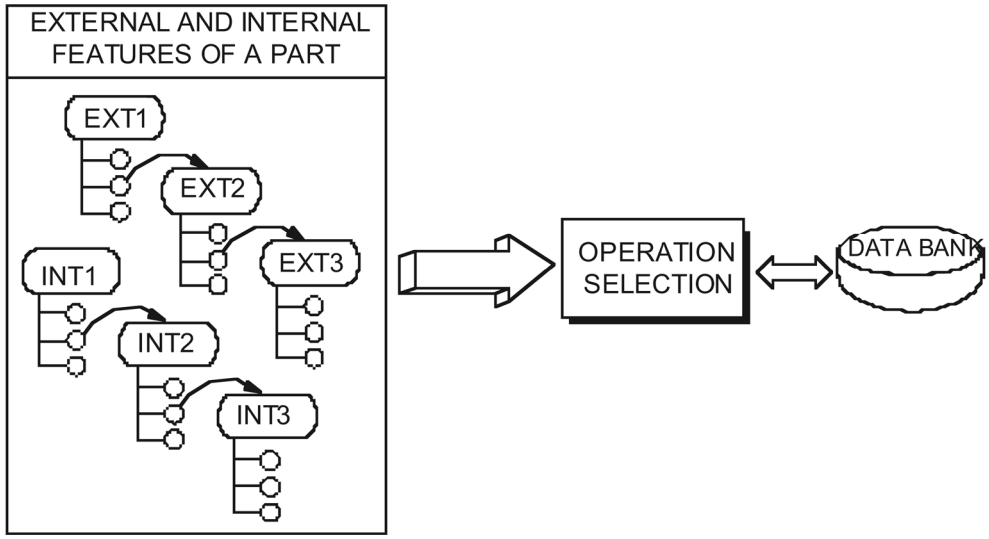


FIGURE 4.20 Operation selection with feature relationship.

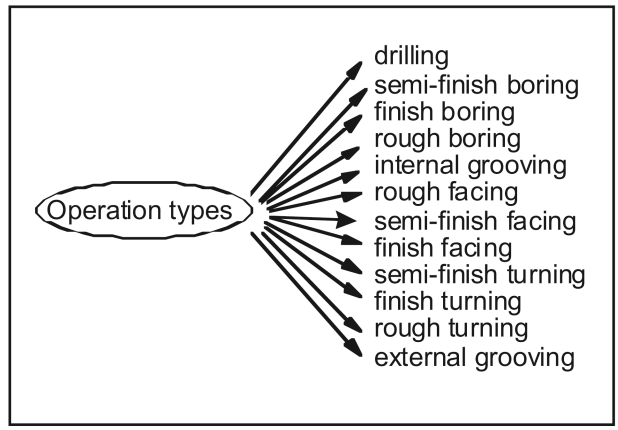


FIGURE 4.21 Types of operation.

From the above discussion, it is clear that a machining feature is linked to one or more types of operation. For the taper feature labeled as EXT6 in Figure 4.7, for example, a rough turning operation followed by a semi-finish turning operation is required, assuming that a good surface finish is not necessary. Thus, the OPERATIONS slot value for EXT6 must be a multi-valued type containing both of these operations. Though this may seem to be a fairly straightforward procedure, an interdependent situation may arise, e.g., to produce a hole with a groove and a chamfer as shown in Figure 4.22. In such a case, drilling, chamfering, and grooving may be sequenced. Drilling is the only operation assigned to produce the hole if the tooling capability is able to meet the desired surface and tolerance requirements. The system, however, will check for drill size availability and perform the necessary changes in the allocation of operations if the correct drill size cannot be found. If so, boring operations consisting of roughing and finishing may have to be preceded by drilling. The initial hole size must be checked for tool collision before any boring operation is assigned.

Tooling capability parameters must be considered in the selection of operations in order to make reasonable plans. They may be obtained from handbooks which provide useful guidelines. Shop floor

© 2001 by CRC Press LLC

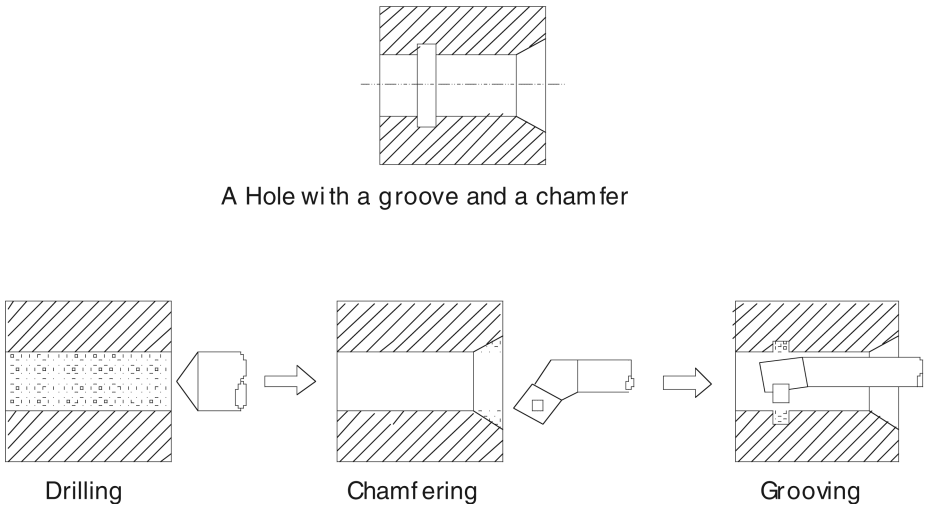


FIGURE 4.22 An example of operation selection.

knowledge may be provided, but is usually proprietary. An example of drilling capability, as shown in Figure 4.23 is dependent on the type of tooling. In general, cemented-carbide drills produce a better surface finish and a better hole tolerance than HSS drills. Normally it is possible to achieve an average surface roughness of R_a 2 to 4 μm and a tolerance of IT10 for delta drills. The straightness of the hole will vary, depending upon the conditions. A tolerance of 0.2 mm per meter of drilled length can be expected with a rotating workpiece. These parameters reflect the capability of current tools. A simplified LISP function for delta drills is written as:

```
; drilling capability for Ra of 4 μm, hole tolerance
; of IT10 for delta drill
;
(Defun drill-constraint (roughness tolerance diameter type)
   (if (equalp type 'delta)
      (and (>= roughness 4)
      (cond ((and (<= diameter 3.0) (>= tolerance 0.04)) t)
            ((and (<= diameter 6.0) (>= tolerance 0.048)) t)
            ((and (<= diameter 10.0) (>= tolerance 0.058)) t)
            ((and (<= diameter 18.0) (>= tolerance 0.07)) t)
            ((and (<= diameter 20.0) (>= tolerance 0.084)) t)
            (t nil))))
   )
```

The surface roughness and tolerance of a feature, i.e., a hole, are checked against the drilling operation selected by the knowledge base. If the drilling capability parameters are found to be unsuitable, a finishing operation is added to the OPERATIONS slot-name of the feature.

Sequencing of Operations

The operation sequencing knowledge in the present work utilizes the CNC capability in the lathe. The machine is equipped with interpolators which allow the cutter to follow the profiles of the component. When all of the features have been assigned with appropriate operations, the next process is to sort the operation lists using the locations of the features in descending order, starting from a reference point

© 2001 by CRC Press LLC

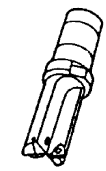
Drills Feature	 Coromant Delta S-drills	 Coromant Delta drills	 T-MAX U solid drills
Drill diameter	2,50 – 12,00 mm	9,50 – 30,00 mm	17,5 – 58 mm
Hole depth	3 × diameter 6 × diameter	3,5 × diameter 5 × diameter	2 × diameter 2,5 × diameter
Hole tolerance	IT 10	IT 9 (3,5 × D) IT 10 (5 × D)	±0,2 mm
Surface finish	R_a 3 μm	R_a 1 – 2 μm (3,5 × D) R_a 2 – 4 μm (5 × D)	R_a 6 – 10 μm
Diameter adjustment	No	No	Non rotating: Max 4,2 mm radial (depending on diameter) Rotating: ±0,15 mm (with eccentric sleeve)
Cutting fluid	External Emulsion or neat oil	Internal Emulsion or neat oil	Internal Emulsion
Shank	Cylindrical	Whistle notch	Whistle notch
Machines	FMS (flexible machining systems), M/C:s, NC and NC-lathes, CNC, automatics, centre and turret lathes and milling machines.		

FIGURE 4.23 The tool capability parameters of various types of drill.

usually taken from the extreme right of the part. The procedure effectively minimizes the number of tool changes by grouping the operations.

Each set of operations is first grouped using the operation types shown in Figure 4.21. The next step is to apply sequencing constraints to the set of operations and produce a suitable sequence for machining. The general machining scheme used is given in the following:

1. Prepare for the internal profile, if any, by removing as much internal stock as possible by a drilling operation.
2. Apply roughing operation to the external stepped profile by a turning/facing canned cycle.
3. Apply finishing/semi-finishing operation to the external stepped profile by a longitudinal/radial canned cycle.
4. Apply grooving operation to all the external grooves.
5. Thread the external threads.
6. Apply rough boring to internal stepped profile by a longitudinal/radial canned cycle.
7. Apply finishing/semi-finishing to internal stepped profile by a longitudinal/radial canned cycle.
8. Apply grooving operations to all internal grooves.
9. Thread all internal threads.
10. Apply part-off operation if workpiece has a free end.

The logical sequence utilizes the processing capability of the CNC lathe, such as the lathe-auto programming cycle.

© 2001 by CRC Press LLC

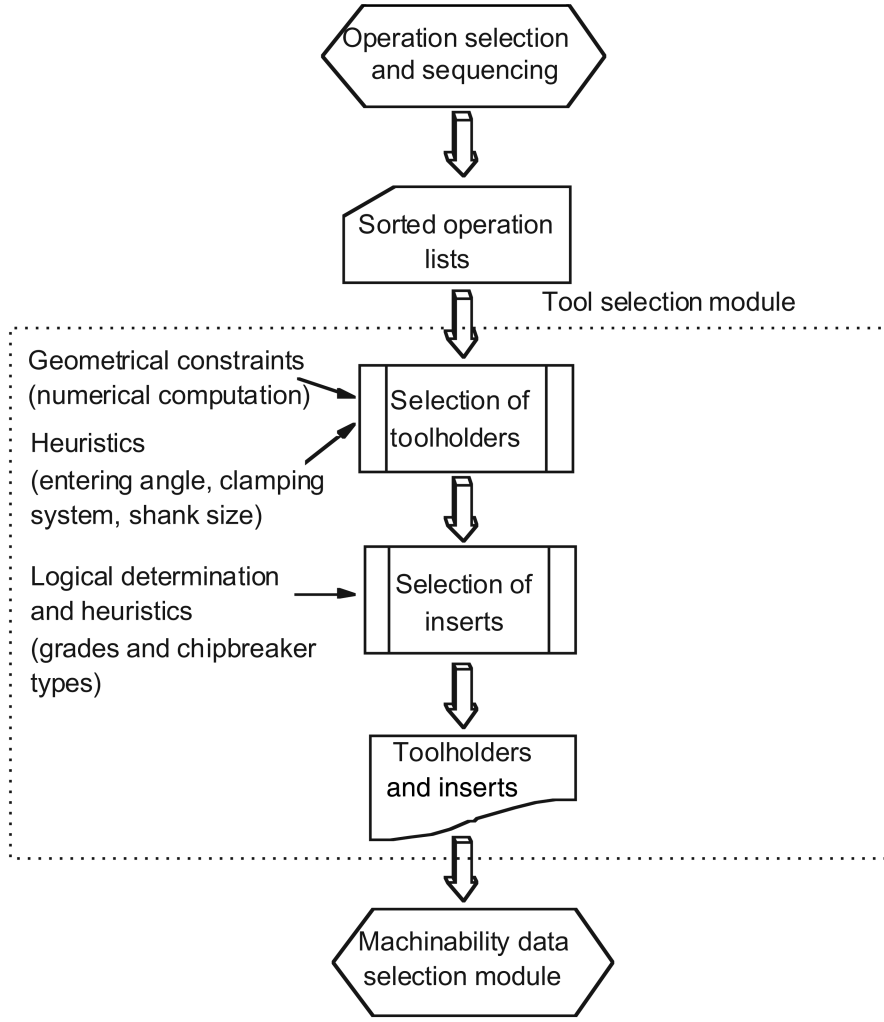

Operation selection and sequencing
Sorted operation lists
Tool selection module
Geometrical constraints (numerical computation)
Heuristics (entering angle, clamping system, shank size)
Selection of toolholders
Logical determination and heuristics (grades and chipbreaker types)
Selection of inserts
Toolholders and inserts
Machinability data selection module

FIGURE 4.24 Elements involved in the tool selection module.

Selection of Tools

An automatic cutting tool selection module has been developed. Figure 4.24 shows the flow chart of the tool selection module used to select a toolholder followed by an appropriate insert for an operation. It is aimed to achieve a good solution quickly and respond to technological changes without recourse to extensive computation effort in economic analysis. Thus, satisfying the operation requirement is a more appropriate approach than optimizing at this stage of the process planning. This is because the selection of a different optimal tool for each feature would result in a requirement for a large number of tools to machine a part. These large number of tools creates a series of problems in tool inventory, scheduling, increased number of tool changes, and, advertently, reduced productivity. The design philosophy of the tool selection module takes into consideration the following requirements:

1. Making use of machining practice. Preference is given to right-hand tools among the tool-types that can be used, namely: left-hand, right-hand, and neutral types. Neutral tools are given the least preference since they produce excessive cutting forces in the radial direction which affect the cylindricity and the diametrical accuracy of the component.

© 2001 by CRC Press LLC

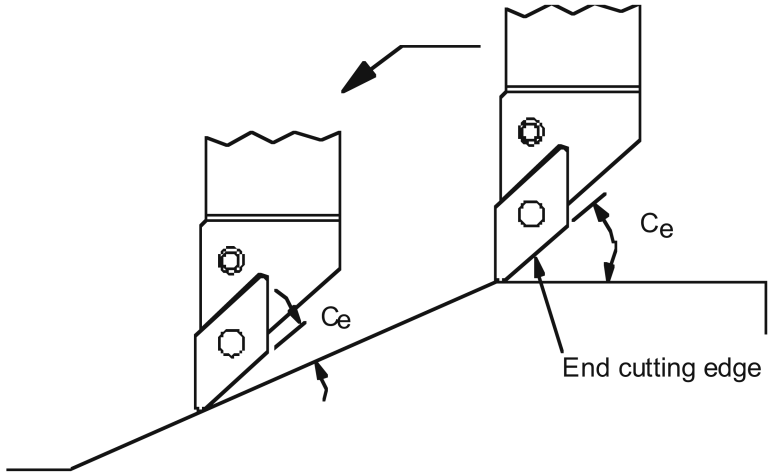


FIGURE 4.25 A reduction in the trailing edge angle due to the change in the feed vector direction.

2. Minimize tool changes. The reduction in tool interruptions effectively economise nonproductive time.
3. Geometric capability and accessing ability. The geometric capability of a toolholder is checked against the operation-type together with the feature's geometry. In boring and turning, left-hand operations will be used if right-hand (system's default) operations are not possible. For example, for a stepped shaft with increasing diameter from the right most, a right-hand tool is used. The diameter of a feature is simply taken using the 'end_point' slot name of the feature. However, if there is a feature with a reducing diameter from its preceding feature (with the exception of grooves), the system will check for adjoining interference between the tool and feature. This phenomenon is affected by the reduction in the effective trailing edge (or referred as end cutting edge) angle, C_e due to the change in the feed vector direction as shown in Figure 4.25. The process of checking for reducing diameter for the next feature is repeated. The method to examine for possible tool collision is illustrated in Figure 4.26. For a taper-type feature (Figure 4.26a), its angle between the end cutting edge and the Z-axis, θ influences the type of right-hand tools that can be used. This is due to the reduction in the trailing edge angle when the feed vector is directed towards the axis of the part. The tools used must be able to perform in-copying, longitudinal turning and out-facing. Some examples of the various tools indicating their respective maximum trailing angles are shown in Figure 4.27. It is found that the maximum taper angle θ of 50° for such a feature can be machined with a right-hand tool. If the value is exceeded, a left-hand operation is used instead.

   For contour-type features, θ is determined from the tangent of the arc to the Z-axis. As for a convex down type (see Figure 4.26b), by taking the worst case situation, the critical position is at the end of the in-copying operation, thus giving:

$$\cos\theta = \frac{X_c - X_e}{R}$$

   On the other hand, for concave down type (see Figure 4.26c), the critical position is at the beginning of the in-copying operation, giving:

$$\cos\theta = \frac{X_c - X_s}{R}$$

© 2001 by CRC Press LLC

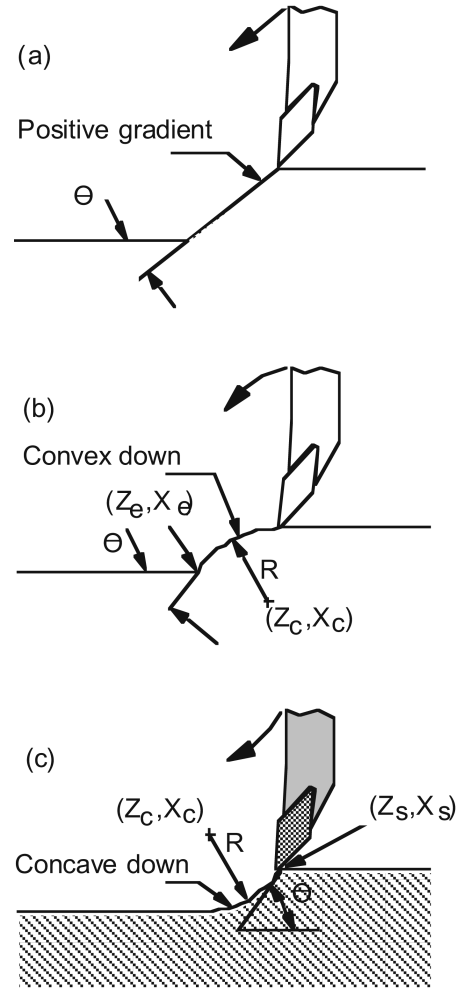


FIGURE 4.26 Interference detection for in-copying operations with right-hand tools.

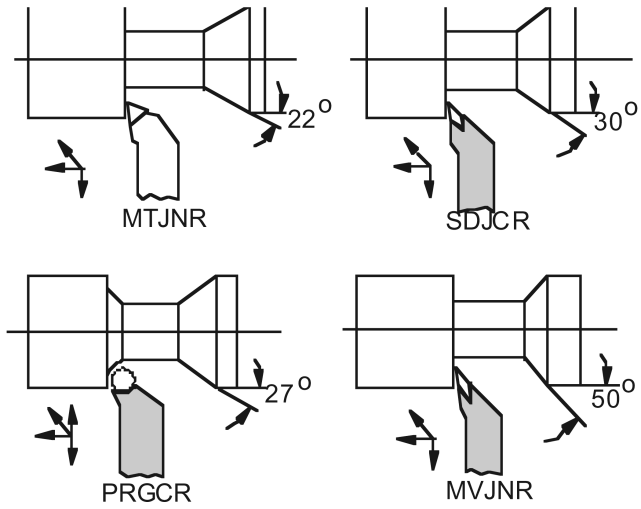


FIGURE 4.27 Right-hand tools with their respective maximum trailing edge angles.

© 2001 by CRC Press LLC

TABLE 4.3 Preference List of Chipbreaker Types and Insert Material Grades for Rough Operations on Low Alloy Steels [Sandvik Coromant, 1990]

Choice	Chipbreaker-type	Insert Material Grade
1st	QR	GC415 (P15)
2nd	MR	GC435 (P35)
3rd	QM	—
4th	UR	—
5th	UM	—

Another critical parameter taken into consideration is the accessing ability of tools in terms of contouring ability and reach ability. Consider a tool attempting to machine arc-type contours in a part. A large tool nose radius machining these feature-types with small radii will simply give unsound solution. In fact, the converse is required, but it is not practical to produce a perfect edge between the end of the in-copying and start of the longitudinal turning, and it serves a limited purpose. The small tool nose radius of 0.4 mm is common and thus an arc radius R of 2.4 mm or more is recommended.

Reach ability is concerned with depth limits which are dependent on the machining processes and tool movements. With a maximum drill length of 60 mm, producing a hole length of 70 mm is quite impossible if the drill is made to approach from one side only. The depth may well be a function of tool length available or limited by the chip removal capabilities of the process.

4. Incorporate heuristics by applying weightage. The merit system is based on the entering angle, the insert clamping system, and the shank size of a toolholder. These factors are particularly important in internal machining. For example, the entering angle should be close to 90° to reduce radial forces which cause tool chatter and vibration during machining. This situation may be aggravated further by the large tool overhang required in deep cuts, or the small hole size requirement which imposes the use of a relatively small shank size toolholder. In order to provide a solution which relates closely to a particular machining situation, the user selects the most important factor amongst the three. For example, in rough boring, a P-type clamping system may be considered to be the most important factor because of its strength in engaging large depth of cut. These factors are obtained from the tool file as described in Section 4.6.1 (i.e., Field Number 13).
5. Inserts relevant to selected toolholders. The selection of inserts for a particular work material and type of operation is based on material grades and chip-breaker types. Table 4.3 gives an illustration of the recommendations in order of priority for rough operations of low alloy steels.

The development of the automatic cutting tool selection module has to consider the above requirements. Efficiency in the tool selection process and flexibility of the module for incremental knowledge building are two important design considerations. The main advantages of the module developed is the utilization of the assigned operations to the machining features and the subsequent intelligent processing of the required information.

The example shown in Figure 4.28 illustrates the procedure for the selection of cutting tools which has been developed to find the most appropriate cutting tool for a given operation.

Step 1

After the stepped profile of the component has been established by the feature recognizer, appropriate operations for each feature have been assigned and sorted, the cutting tool selection module first examines the geometries of all features in an operation list. An assertion pattern named EXT-FEATURES STEPPED-TYPE is asserted into the knowledge base since these geometries of the features (i.e., EXT6, EXT5, EXT4, EXT3, EXT2, and EXT1) are in increasing diameters.

© 2001 by CRC Press LLC

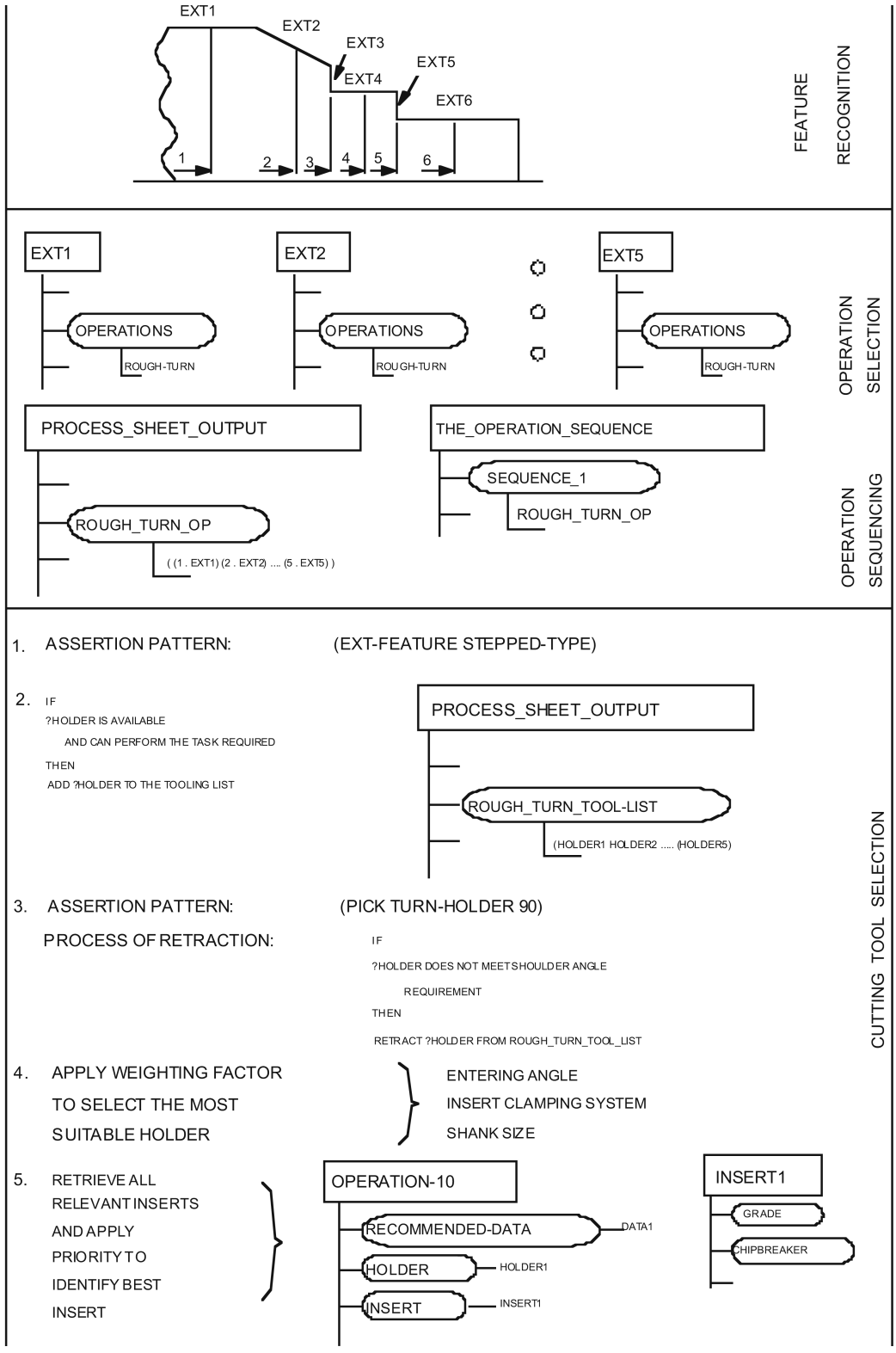


FIGURE 4.28 An example of the cutting tool selection procedure.

© 2001 by CRC Press LLC

Step 2

The geometrical features of each operation are examined for possible constraints on the tooling, such as the suitability of cutter types and the allowable cutter diameter. Applying these constraints to the tooling available for the type of lathe used leads to a list of possible tools being generated for each operation list. A rule to examine the suitability of a tool holder is given as:

```
(define-rule suitable-tools                    ; rule name
   ()                                          ; options if needed
   (ext-features stepped-type)                 ; pattern
   (instance ?hol is holder with function ?fcn)  ; toolholder ?hol is found
   (tool-can-perform? '(axial-turn) ?t)        ; to match required function for
                                               axial-turn
   then
   (instance process_sheet_output is)          ; assert slot-value of the toolholder
   process_sheet with rough_turn_tool_list ?ins))  found and to the
                                               process_sheet_output
```

All the tool holders which performs an axial-turn function will be asserted to the tooling list in the PROCESS_SHEET_OUTPUT instance.

Step 3

This step is required to check on the shoulder angle requirements for each feature. In this example, the 90° shoulder angle requirement is the critical constraint (see Figure 4.28). Thus, a tool holder with an entering angle of more than 90° prevents adjoining interference. An assertion pattern PICK TURN-TOOL 90 is defined into the knowledge base. The RETRACTING-TOOLS rule, as shown below, performs a retraction process (on the list of tool holders obtained in Step 2) by removing tool holders which do not meet the assertion pattern:

```
(define-rule retracting-tools                  ; rule name
   ()                                          ; options if needed
   (instance process_sheet_output is process_sheet  ; instance with a toolholder ?hol
      with rough_turn_tool_list ?hol)
   (instance ?hol is holder                    ; the toolholder with entering
      with entering_ang ?deg)                  ; angle of ?deg
   (pick rough-turn-tool ?shoulder)            ; pattern is assertion base
   (>= ?shoulder ?deg                          ; check for condition
then
   (retract (process_sheet_output              ; retract the toolholder from the tool list
      rough_turn_tool_list ?hol))
```

Step 4

The fourth step involves the selection of the most suitable toolholder amongst the list of toolholders which have satisfied the functional and geometrical constraints. Production rules can be incorporated, such as (a) choose the P-type clamping system for rough turn, and (b) use positive rake holders in situations where vibration tendencies, poor rigidity setup, or unfavourable conditions occur. Where necessary, the user will be asked to indicate their relative importance which closely relates to a particular machining situation.

Step 5

When the selection of tool holders has been completed, the final step is to select appropriate inserts. The goal is to retrieve all applicable insert shape for the toolholder and subsequently identify the best insert

© 2001 by CRC Press LLC

for the operation based on its individual merit. Two factors are important in an automated machining environment: first, the type of chipbreaker, and second, the material grade of the insert which has a significant influence on the tool performance. The guidelines in this part of the procedure have been adopted from a tool manufacturer's machining data [Sandvik Automation, 1989]. An example of guidelines in the form of a preference list is given in Table 4.3. In this case, an insert of chipbreaker-type 'QR' and insert material grade 'GC415' has the highest preference.

Selection of Machining Conditions

The selection of machining conditions for the IMS is from the general methodology for machinability data selection [Yeo, Rahman, and Wong, 1989]. The cutting tool chosen, type of operation, and work material used, as well as the machine and controller type, will subsequently influence the machining conditions because of their interdependency. The procedure for selection of machining conditions is described as follows:

Step 1

This step is to retrieve a recommended range of feed and depth of cut which provide good chip control. This is dependent on the work material group and chip-breaker type. For example: for AISI 1045 and chip-breaker type 'QM', the feed range of 0.2 to 0.5 mm/rev and the depth of cut ranging from 0.25 to 2.0 mm are suggested.

Step 2

This step requires the input of feed and depth of cut. A procedure is used to recommend a maximum allowable feed value which is dependent on the insert nose radius, the average surface roughness required, and the operation type. The depth of cut may be recommended if an appropriate rule for the conditions is matched, for example:

```
(define-rule recommend-depth-of-cut          ; rule name
  ()                                         ; options if needed
  (instance the-part-info is part-info       ; part description of
        with material-description carbon-steel ; this type and
        with hardness 150                    ; hardness
         with gen-surface-roughness 1        ; surface-roughness
         with tolerance 0.01)                ; and diametral tolerance
  then                                       ; if true then
  (evaluate (recommend depth-of-cut 0.5)     ; recommend 0.5 mm depth of cut
```

Step 3

This step computes the recommended cutting speed based on the tool life of 15 min, which is the system's default value. The power required and cutting time are also computed. This is done by:

```
(send-msg 'the-operation-data :compute-cutting-data)
```

Thus, a 'send-msg' function invokes the handler (i.e., procedural attachment) named ':compute-cutting-data' to perform numerical computation for the instance named 'the-operation-data.' User modification of the feed and speed may be necessary, i.e., if the power requirement has exceeded the machine power. The computation of the tool life, cutting time, and power required is repeated using the procedure described by Yeo, Wong, and Rahman [1991].

Step 4

This step checks for relevant production rules representing shop floor heuristics that matches the current situation. An example of a production rule to use an alternative set of cutting data is as follows:

© 2001 by CRC Press LLC

```
(Define-rule shop-floor-rule-1                          ; rule name
    (:explanation-string "Modification of cutting data")  ; explanation
    (instance the-part-info is part-info
      with material-code 1045                            ; part material type and
      with material-hardness 220)                        ; hardness
    (instance the-operation-data is operation-data
      with holder ?hol                                   ; the toolholder
      with insert ?ins)                                  ; and insert used
    (instance ?hol is holder
      with identification_code PCLNR_2525M_12)           ; the holder and insert
    (instance ?ins is insert                             ; matches the tool
      with identification_code CNMG_12_04_08-QF_GC415    ; identification
    then
    (evaluate (use-shop-floor 1045-200))                 ; use shop floor data
```

Optimization of machining conditions of multi-pass lathe operations can also be incorporated [Yeo, 1995]. Depending on the availability of data sets, the solutions obtained is easily programmed utilizing CNC processing capabilities.

Generation of Tool Paths

The results obtained from (a), (b), (c), (d), and (e) as shown in Figure 4.2 are used for CNC part programming. In order to maintain consistency in integration capability, a CNC module for the generation of tool paths has been developed as shown in Figure 4.29. Production rules and object-oriented programming techniques have been used in the module. The basis of using these techniques is due to their modular building structure. Preliminary work has shown that the techniques can be feasibly deployed for part programming [Kramer, 1989]. The strategy for tool path generation is to use the built-in processing ability of the OSP5020L CNC system [Okuma Corporation, 1991]. The lathe auto-programming (LAP) functions, such as the G85 code for longitudinal turning, eliminate programming of the intermediate geometrical coordinates which requires lengthy blocks of program. The G85 code may be used for downward cuts, i.e., turning decreasing diameters in the direction of tool travel as shown in Figure 4.30.

A frame named 'NC-FRAME' is used to form the structure needed to produce all the individual blocks of controller code, i.e., tool path for each operation (see Figure 4.29). The order of generating the codes is based on the information given in 'THE_OPERATION_SEQUENCE' instance. The tooling pertaining to the operations concerned are found in the instances of 'OPERATION-DATA' frame, while the list of related features are obtained from the 'PROCESS_SHEET_OUTPUT' instance. Seven rules relating to the types of operation are used to control the generation of tool paths for the Okuma lathe. An example of a 'nc-for-facing' rule is given by:

```
(Define-rule nc-for-facing                              ; rule name
  ()                                                    ; options if needed
  (instance ?op1 is operation-data                      ; system checks for
    with operation-in-sequence rough_facing_op)         ; facing operations
  (or (instance ?op2 is operation-data
    with operation-in-sequence semi_finish_facing_op)
  (instance ?op2 is operation-data
    with operation-in-sequence finish_facing_op))
  (generate-nc-data lap-facing)                         ; assertion pattern
  (bind ?instance (gentemp ?nc-facing-))                ; creates a "starter" instance
  then                                                  ; if the above is true then
  (send-msg ?op1 :turret)                               ; request for tool
  (send-msg ?op2 :turret)                               ; turret positions
  (retract (generate-nc-data lap-facing)                ; retract assertion base
  (instance ?instance is nc-frame)                      ; creates a starter instance
  (send-msg ?instance :fill-slots 'lap-facing ?op1 ?op2))  ; generates tool path
```

© 2001 by CRC Press LLC

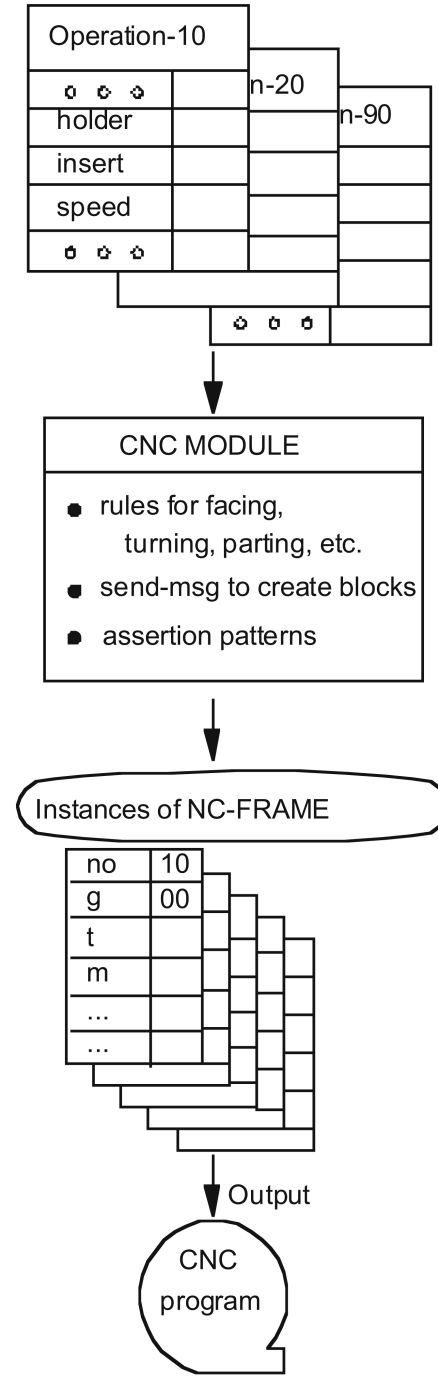


FIGURE 4.29 Methodology used for the generation of tool paths.

The above rule is fired if the operation-type and the assertion pattern are matched. The antecedent part will then perform.

1. Input tool turret positions.
2. Retract the assertion pattern for the operation type.
3. Create instances in sequence; each instance represents a block of program.

© 2001 by CRC Press LLC

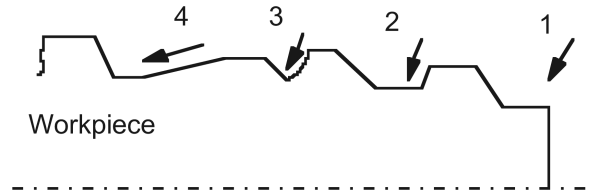


FIGURE 4.30 A part contour indicating four downward cuts.

Each of the seven rules have been developed similar to the 'nc-for-facing' rule. When all tool paths have been generated, the CNC program file is output using a 'output-stream' function. This file is then downloaded to the CNC controller. The results obtained from (a), (b), (c), (d), and (e) as shown in Figure 4.2 are presented in a process sheet to aid a CNC operator. Minimal adjustments are required for tool setting, tool nose compensation, and program stop function due to limited turret capacity before actual machining.

The developed CNC module has been able to (a) use production rules and object-oriented programming techniques to provide a flexible and modular approach, (b) integrate with the machining planning activities, and (c) reduce part programming effort. Thus, the human intervention in part programming is significantly reduced.

4.8 Concluding Remarks

An integrated machining system is one of the essential component in automated manufacturing environments. The feature recognition, the selection of operations, the sequencing of operations, the selection of tools, the selection of machining conditions, and the generation of tool paths have been considered by the IMS. A frame-based expert system building tool involving production rules and object-oriented programming techniques has been used in this research and found to be well suited for handling these activities. The developed IMS is capable of generating a production plan of actions and CNC programs for machining rotationally symmetric parts.

In the system, all modules are integrated into a coherent whole, and various strategies are incorporated to enhance the level of expertise provided by the system. The size and quality of the knowledge base of the IMS has provided sufficient a performance level that constitute a knowledge-based system. The system's performance can be easily enhanced further by encoding rules to improve future decision making.

The various details discussed in this chapter are summarized below:

1. The architecture of the integrated knowledge-based machining system and general description of the design philosophy were presented.
2. A window-based user interface was used to provide friendly interactions between the user and the system.
3. A feature recognizer to transform and interprete geometrical models in DXF format has been developed. A frame-based approach to formulate a lattice structure of machining features of a rotationally symmetric part has provided an efficient means of representing a generic model. The organization of the lattice structure is suitable for efficient pattern matching of rules. The methodology has been used for a wide range of machining features in a part and provides the flexibility to change feature definitions with minimal effort.
4. An interface for data acquisition of factual knowledge from external data base files has been elaborated. The data structure and data contents of cutting tools, work materials, etc., have been presented. Application programs for data retrieval and data search have been developed.
5. A methodology using feature relationships for the selection of operations has been adopted. Tooling types, tooling availability, and tooling capability parameters are main consideration in assigning appropriate operations to the features.
6. Knowledge for the sequencing of operations has been formalized. The logical sequence utilizing the processing capability of the CNC lathe has been highlighted.

© 2001 by CRC Press LLC

7. An automatic tool selection module to meet the functional and heuristic aspects of tooling has been described. Efficiency in the tool selection process and flexibility of the module for incremental knowledge building are two important design considerations. The main advantages of the module developed is the utilization of the assigned operations to the machining features and the subsequent intelligent processing of the required information. Five steps governing the selection of tool holders and inserts have been discussed.
8. Four steps involved in the selection of machining conditions procedure have been presented and detailed. The methodology used is based on the data base approach with shop floor heuristic.
9. A CNC module has been developed. Production rules and object-oriented programming techniques have been used to provide a flexible and modular approach. The module has been integrated with other machining planning activities. Part programming effort and intensity of human intervention are significantly reduced. The developed IMS has been able to satisfy the following important considerations.

    - Modularity, expandability, and adaptability to different operating environments.
    - Interactive prompting routines that lead a user to retrieve information in an orderly manner.
    - Data base of updatable files containing standard information, proprietary information, and product-facility oriented data of combinations of cutting tools, work materials, etc.

References

Evershiem, W., Jacobs, S., and Wienand, L., 1987, Structure and application of a universal company-independent data bank for tools, *Annals of CIRP,* 36(1), 321–325.

Gold Hill Computers Inc., 1989, *Goldworks III Man.,* Chestnut Hill, MA.

Hinduja, S. and Huang, H., 1989, Automated detemination of work-holding parameters for turned components, *Proc. Inst. of Mech. Engineers,* Part B, 203, 101–112.

Kramer, W., 1989, *Object oriented NC Programming with CAD/CAM,* SME Tech. Pap., (MS89-694), 24–12 to 24–28.

Okuma Corporation, 1991, Programming Man., 10th Ed., Publ. (3277-E-R2), Japan.

Sandvik Automation, 1989, COROPLAN, 2.22.

Sandvik Coromant, 1990, *Turning Tools and Inserts,* C-1000:3-ENG.

Weill, R., Spur, G., and Eversheim, W., 1982, Survey of computer-aided process planning system, *Annals of the CIRP,* 31(2), 539–551.

Yeo, S.H., 1995, A multipass optimization strategy for CNC lathe operations, *Int. J. Prod. Econ.,* 40, 209–218.

Yeo, S.H., 1995, Knowledge-based feature recognizer for machining, *Comput. Integrated Manufacturing Syst.,* 7(1), 29–37.

Yeo, S.H., Rahman, M., and Wong, Y.S., 1989, Towards enhancement of machinability data by multiple regression, *J. of Mech. Working Technol.,* 1(1), 85–99.

Yeo, S.H., Rahman, M., and Wong, Y.S., 1990, A frame-based approach for the making of holes in turned parts and its further development, *J. Mater. Processing Tech.,* 23(2), 149–162.

Yeo, S.H., Rahman, M., and Wong, Y.S., 1991, Integrated knowledge-based machining system for rotational parts, *Int. J. Prod. Res.,* 29(4), 271–281.

Yeo, S.H., Wong, Y.S., and Rahman, M., 1991, Knowledge-based systems in the machining domain, *Int. J. Adv. Manufacturing Tech.,* 6(1), 35–44.

Zdeblick, W.J., 1983, Computer-aided process and operation planning, *Realizing Your Factory of the Future-Today*, Metcut Res. Assoc., 5.1–5.19, Cincinnati.

© 2001 by CRC Press LLC

5

Intelligent Set-Up Planning Systems for Parts Production in Manufacturing Systems

Abstract
Keywords
5.1 Introduction
5.2 Machine and Component Levels Set-Up Planning
5.3 Two Viewpoints of Set-Up Planning
The Machining Viewpoint • The Fixturing Viewpoint
5.4 Factors and Constraints in Set-Up Planning
Approach Directions of Features • Geometrical Relationships • Design Specifications • Machining Requirements • Fixturing Requirements
5.5 Features Interactions in Set-Up Planning
5.6 Artificial Intelligence and Set-Up Planning
5.7 Open Research Issues
Set-Up Validity and Optimization • Set-Up Planning and Product Design • Set-Up Planning and Shop Floor Control
5.8 Summary
References

S.K. Ong
National University of Singapore

Andrew Y.C. Nee
National University of Singapore

Abstract

Set-up planning is the task of organizing and determining a sequence necessary to make the features in certain workpiece orientations. It is a pivotal step in automated process planning as it greatly influences machine and tool selections, machining sequences, and fixture configurations. This paper reviews the methodologies and techniques in the development of computer-aided set-up planning systems. It examines the status of, and suggests some future directions for, research efforts in computer-aided set-up planning—an area that is of premier importance in integrated manufacturing planning. Most of the published papers have implemented set-up planning from either one of the following micro-viewpoints: the machining or the fixturing viewpoint. This paper examines these two different viewpoints of set-up planning and discusses the merits and limitations of the work done in these areas. The set-up planning problem is analysed at two levels: the machine and the component levels, to determine the essential factors and constraints in set-up planning. Recent work in computer-aided set-up planning has attempted to simultaneously adopt these two views, and to achieve the integration with design evaluations and shop

© 2001 by CRC Press LLC

floor control systems. This paper presents these recent developments and discusses a few open issues in set-up planning.

Keywords

Process planning, fixture planning, set-up planning, features

5.1 Introduction

Since the beginning of the 1980s, manufacturing planning has been recognized by both academia and industry to be vital in achieving the ultimate goal of unmanned and integrated factories of the future. Planning is an intrinsic part of intelligent behaviour, often performed subconsciously by human beings. It can be viewed as the activity of devising means to achieve desired goals under given constraints and limited resources [Ham and Lu, 1988].

Manufacturing planning is the process of coordinating the various activities in the design and manufacturing processes. It is traditionally performed in two stages that communicate through an interface called a process plan, as shown in Figure 5.1. In the first stage, an operations planner and a fixture planner collaborate to produce a process plan, which is usually a concise document specifying a sequence of

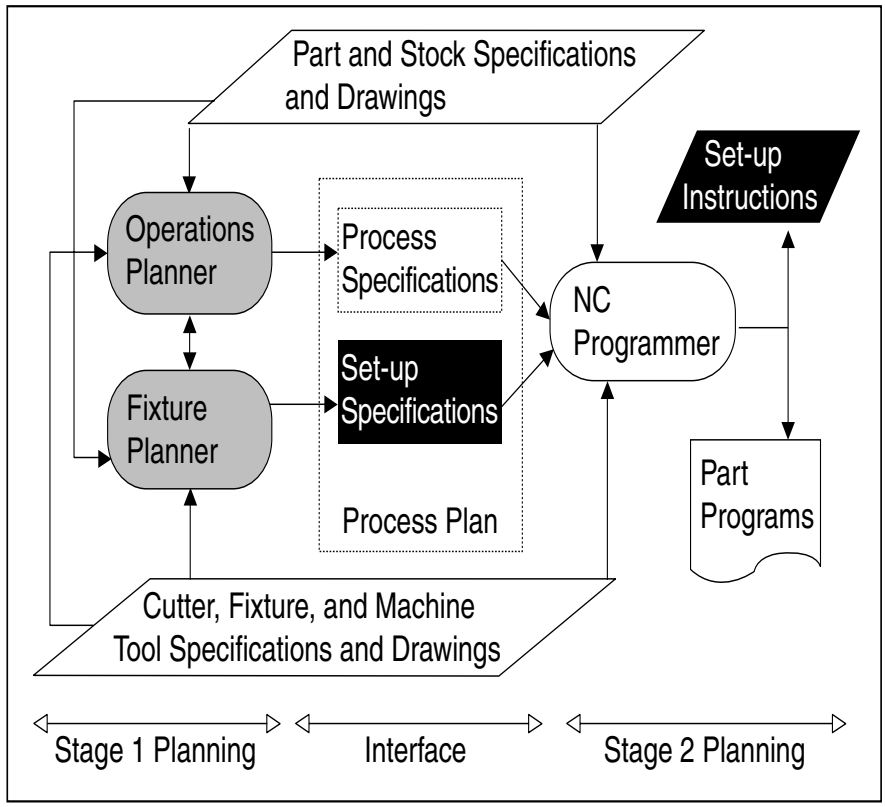


FIGURE 5.1 Traditional two-stage approach to manufacturing planning.

© 2001 by CRC Press LLC

operations. In the second stage, a numerical control (NC) programmer generates detailed specifications for each operation in the plan. These are typically part programs for NC machines plus set-up instructions for human operators.

Currently, manufacturing planning has generally been viewed as a hierarchically structured activity for achieving factory integration through bridging design and manufacturing. Most current planning systems, however, are not well integrated. Current efforts tend to focus on specific planning functions for specific workpieces [ElMaraghy, 1993; Bullinger, Warnecke, and Lentes, 1986]. For example, one group of research studies the extraction and representation of part features from solid models of workpieces [Salomons, Van Houten, and Kals, 1993; Shah, 1988; Shah, 1991], and features modeling and conversion [Bronsvoort, and Jansen, 1993; Shah, Mäntylä, and Nau, 1994; Shah, 1992]. A second group concentrates on Stage-I planning in Figure 5.1, i.e., the selection and sequencing of operations for machining the features on a part, often using artificial intelligence methods [Weill, Spur, and Eversheim, 1982; Ham and Lu, 1988; Zhang, and Alting, 1994]. Another group covers systems that support the programming of machine tools and contain some automatic operations planning facilities. These micro-viewpoint planning approaches produce computer-aided systems that perform individual tasks in isolation from other planning activities. These computer systems tend to focus on a narrow range of activities that severely limit their applicability in practice. For example, many computer-aided process planning (CAPP) systems that have been reported to date are aiming at generating the machining sequences of features and the selection of machining operations [Alting and Zhang, 1989; Zhang, and Alting, 1994]. The same can be said of the many computer-aided fixture planning (CAFP) systems that have been developed to automate the fixture design and planning process [Hargrove and Kusiak, 1994; Nee and Senthil Kumar, 1991; Trappey and Liu, 1990]. Other areas such as sheet metal forming processes and plastic moulding processes are all challenging domains for planning systems, but have received only minor attention so far [ElMaraghy, 1993].

Broadly speaking, the entire process planning domain for the machining environment can be divided into three levels, namely (a) operations planning, (b) set-up planning, and (c) fixture planning [Sood, Wright, and MacFarlare, 1993]. The most important of these is set-up planning because almost all the processes in machining are set-up dependent, as illustrated in Figure 5.2. The set-up process has been estimated to make up to 60% of the production time on a CNC turning center, and greater than 60% for a CNC machining center [Venjara, 1996]. Thus, the reduction of the set-up time and cost of a set-up plan is vital for achieving efficient production. Set-up planning is a link to integrate operations planning with fixture planning as both activities can be considered concurrently [Ong and Nee, 1994a].

An automated process planning system should strictly encompass all three levels of planning. A critical review shows that many of these systems do not address the entire planning problem, but instead concentrate on the automation of one of these planning functions. Many of the reported CAPP systems can solve the first planning function successfully, which is operations planning. These systems perform functions such as selecting the least cost operation for each feature on a workpiece, determining the feeds, speeds, and processes for generating the individual feature, and sequencing the operations for generating these features [Westhoven et al., 1992; Züst and Taiber, 1990; Nevrinceanu and Donath, 1987a; Nevrinceanu, 1987b]. Another group of micro-viewpoint CAFP systems solves the third planning function, which is fixture planning. These CAFP systems plan the locating, clamping, and supporting positions, and design the fixture configurations to hold workpieces during the machining operations [Hargrove and Kusiak, 1994; Nee, and Senthill Kumar, 1991; Trappey, and Liu, 1990]. Both groups of micro-viewpoint systems do not address planning at the higher level, i.e., set-up planning [Ong and Nee, 1994b], although a few of them do perform a certain level of set-up planning in their implementation.

The premier set-up planning problem is automatic design of set-ups and set-up sequences. As mentioned earlier, set-up planning is a function of process planning that has been largely neglected by researchers working on CAPP. In this paper, techniques that have been applied to computer-aided set-up planning are discussed. This paper also examines the status of, and suggests some future directions for, research efforts in computer-aided set-up planning. Tables 5.1 and 5.2 give respectively the CAPP and CAFP systems that have incorporated set-up planning in their implementation. Table 5.3 gives a list of computer-aided set-up planning systems.

© 2001 by CRC Press LLC

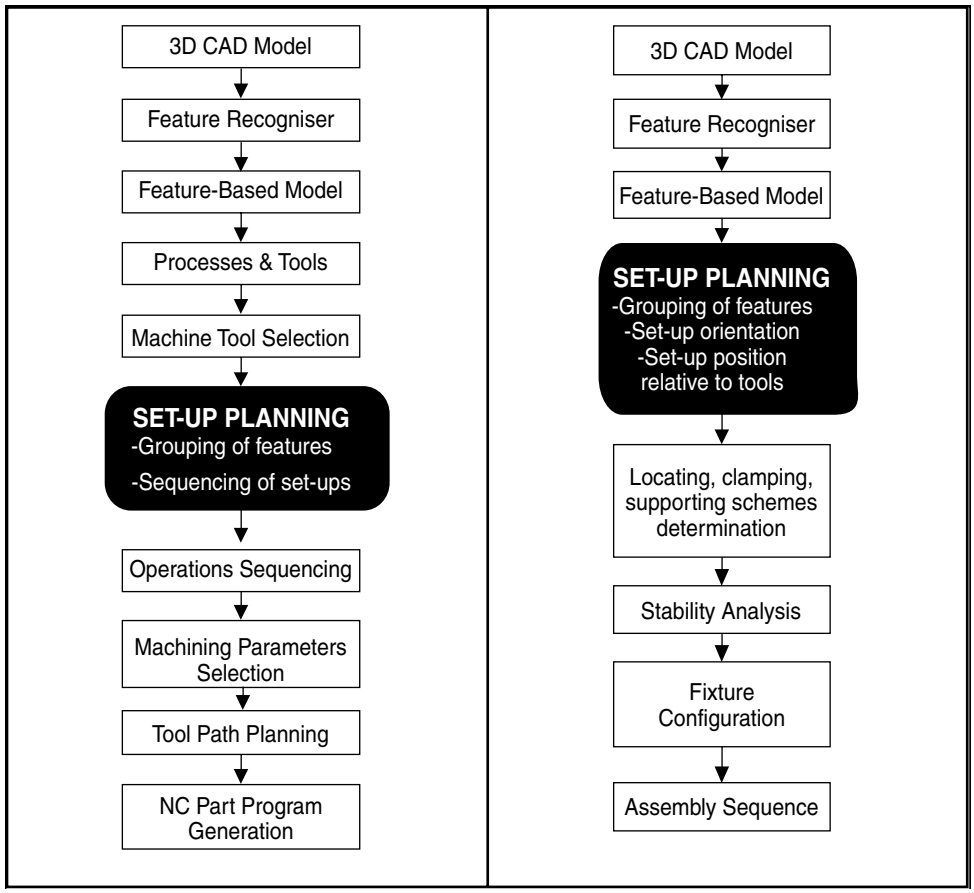


FIGURE 5.2 General operations planning and fixture planning frameworks.

5.2 Machine and Component Levels Set-Up Planning

Set-up planning can be split into (a) *component set-up level*, which considers the set-up planning problem in relation to a single component, and (b) *machine batch set-up level*, which considers the batch and machine requirements on the machine tools.

Component set-up level planning is concerned with identifying an ordered sequence of set-ups for a workpiece where each set-up contains (a) regions to be machined, (b) operations to be performed, (c) possible tools and processing parameters for each operation, (d) regions for location, (e) regions for clamping, and (f) orientation of the set-up. Majority of the systems listed in Tables 5.1, 5.2, and 5.3 tackle the set-up planning problem at this level. Factors and constraints that are of importance at this level are essentially the design specifications of the features on a workpiece, the geometry and topology of the workpiece, tolerance values, etc. These constraints are analysed to determine the relations between the features on the workpiece for formulating the set-ups that are needed to machine the required features and the sequence of generating these features. This level of set-up planning has a very close link with design evaluation and cost analysis of workpieces [Ong and Nee, 1994a]. The features on a workpiece can be redesigned by analysing the set-up plans so that fewer set-ups will be needed to machine the workpiece [Hayes, Desa, and Wright, 1989; Mäntylä, Opas, and Puhakka, 1989; Ong, and Nee,1994a], thus reducing the cost of the design. Table 5.4 gives two design systems that have incorporated set-up planning during the design evaluation process based on this concept.

Hayes, Desa, and Wright, [1989] reported an iterative redesign methodology as a means of using set-up planning information to find ways of reducing the cost of a design by combining and/or eliminating set-ups.

© 2001 by CRC Press LLC

© 2001 by CRC Press LLC

TABLE 5.1 CAPP Systems with Set-up Planning

Authors	Functions/ Name	Viewpoint	Fixturing System	Parts	Criteria	Solid-model Integration	Machining Environment	Reasoning Techniques	Level of Set-up Planning
Armstrong et al. 1984.	automatic NC code generation	machining		prismatic	1. maximum material removal directions 2. tool cutting paths	yes; PADL-1; spatially ordered representation	3-axis vertical machining centre	rules; features grouping based on ADs; set-up sequencing based on criterion	component level; set-up forming and sequencing
Chan and Voelcker, 1986.	process planning	fixturing	machining vise	prismatic	1. part positioning requirements 2. part clamping requirements	yes; PADL-2; CSG solid models	3-axis vertical machining centre	rules	component level; interactive set-up planning
Joshi et al. 1988.	process planning	machining		prismatic	1. geometrical relations 2. spindle axis directions 3. precedence relations	yes; BREP solid models		rules; features clustering; set-ups sequencing based on precedence relations	component level; set-up forming and sequencing
Bond and Chang, 1988.	process planning	machining		prismatic	1. machines requirements 2. fixturing requirements 3. spatial relations	yes; UCLA Intelligent CAD models		rules; features clustering	machine level; set-up forming
Mantyla and Opas, 1988; Mantyla et al. 1989.	process planning - HUTCAPP	machining		prismatic	1. machining directions 2. cutting tools	no; feature-based models	3-axis vertical machining centre	rules; features grouping based on ADs; set-ups sequencing based on number of cuts in each set-up	component level; set-up forming and sequencing
Bell and Young, 1989.	process planning - Machine Planner	machining	machining vise	$2\frac{1}{2}$ D prismatic	1. critical tolerances 2. maximum material removal 3. clamping strategy	yes; CSG solid models	3-axis vertical machining centre	rules; features clustering based on ADs	component & machine levels; set-up forming and sequencing

(*continued*)

© 2001 by CRC Press LLC

TABLE 5.1 CAPP Systems with Set-up Planning (Continued)

Authors	Functions/ Name	Viewpoint	Fixturing System	Parts	Criteria	Solid-model Integration	Machining Environment	Reasoning Techniques	Level of Set-up Planning
Joneja and Chang, 1989; Anderson and Chang, 1990; Joneja and Chang, 1991.	process planning - QTC	fixturing	machining vise	prismatic	1. geometrical relations 2. tolerance constraints 3. ADs of features 4. machining precedence relations 5. fixturing requirements	yes; BREP solid models; TWIN solid modeler		rules; features clustering based on ADs; set-up sequencing based on precedence relations	component level; set-up forming and sequencing
Gindy and Ratchev, 1991.	process planning - GENPLAN	machining		prismatic	1. ADs of features 2. precedence relations of features 3. maximum number of features	no; feature-based models	3-axis vertical machining centre	rules; features clustering based on ADs	component level; set-up forming and sequencing
Mayer et al, 1992.	process planning - IMPA	machining	machining vise	prismatic	1. tool directions 2. maximum material removal 3. clamping requirements 4. interference checks	yes; interface via IGES file; BSPT data structure	3-axis vertical machining centre	rules; features clustering; breadth-first search strategy	component level; set-up forming and sequencing
Warnecke and Muthsam, 1992; Muthsam and Mayer, 1990.	process planning - EXPLAN	machining	machining vise	prismatic	1. obligatory machining sequence 2. spindle directions 3. dimensional tolerances 4. clamping requirements	IGES 3D interface; conversion to IAOGraphs	3-axis vertical and horizontal machines; boring and drilling machines	rules; features clustering based on ADs; set-up sequencing based on limiting conditions set-ups	component level; set-up forming and sequencing
Delbressine et al., 1993.	process planning - IDM	machining	modular fixture elements	prismatic	1. tolerance specifications 2. geometric reachability of features with respect to tools	yes; hybrid of BREP and CSG solid models	3-axis vertical machining centre	rules; merging of tolerance and precedence graphs	component level; set-up forming and sequencing

© 2001 by CRC Press LLC

Opas, 1993; Opas et al., 1994.	process planning - MCOES	machining	modular fixture elements	prismatic	1. machining directions of features 2. tolerance specifications	yes; BREP GWB modeler	3-axis vertical machining centre	rules	component level; interactive set-up planning
Gu and Zhang, 1993.	process planning - OOPPS	fixturing	machining vise	prismatic	1. machine requirement 2. fixturing requirement 3. features accessibility 4. maximum features machining 5. tolerance specifications	yes; Autosolid solid modeler	3-axis vertical machining centre	rules; recursive approach	machine and component levels; set-up forming and sequencing
Jung and Lee, 1994.	process planning	machining and fixturing	machining vise	prismatic	1. datum requirements 2. ADs of features 3. set-up interference 4. clamping requirements	no; feature-based models		rules; branch & bound optimisation	component level; set-up forming and sequencing
Hwang and Miller, 1995.	process planning	machining		prismatic	1. tolerance requirements 2. geometric reasoning	no; feature-based models		blackboard architecture; backtracking	component level; set-up forming and sequencing

© 2001 by CRC Press LLC

TABLE 5.2 CAFP Systems with Set-up Planning

Authors	Functions/ Name	Viewpoint	Fixturing System	Parts	Criteria	Solid-model Integration	Machining Environment	Reasoning Techniques	Level of Set-up Planning
Englert and Wright, 1986.	fixture planning - Expert Machinist	fixturing	machining vise or toe clamps	prismatic	1. machining practices 2. ADs of features 3. maximum number of features	no; CML language	3-axis vertical machining centre	rules; tables of cuts and orientations	component level; set-up forming and sequencing
Young and Bell, 1991.	fixture planning	machining	machining vise	$2\frac{1}{2}$ D prismatic	1. critical tolerances 2. maximum material removal 3. clamping strategy	yes; spatially divided solid models	3-axis vertical machining centre	rules; features clustering based on ADs	component level; set-up forming and sequencing
Boerma and Kals, 1988; Boerma and Kals, 1989; Boerma, 1990.	fixture planning - FIXES	fixturing	modular fixture elements	prismatic	1. tolerance specifications evaluation 2. face orientation of features 3. machine tool directions 4. fixturing requirements 5. most accurate tolerance machined	yes; BREP solid models, GPM	3-axis vertical machining centre	rules; features grouping based on tolerance relations; set-ups sequencing based on criterion	component level; set-up forming and sequencing
Ferreira and Liu, 1988.	fixture planning	fixturing	modular fixture elements	prismatic	1. maximum number of features machining 2. ease of fixturing 3. release of precedence relations 4. dimensional tolerances specifications 5. workpiece stability	yes; BREP solid models; feature-based models	3-axis vertical and horizontal machines; boring and drilling machines	rules; features clustering; generate-and-evaluate strategy	component level; set-up forming
Sakurai, 1990; Sakurai and Gossard, 1991; Sakurai, 1992.	fixture planning	fixturing	modular fixture elements	prismatic	1. datum requirements 2. maximum number of features first 3. ADs of features 4. clamping requirements	yes	3-axis vertical machining centre	rules; back-tracking strategy with kinematics analysis	component level; set-up forming and sequencing
Lee et al. 1991; Kambhampati et al. 1993.	process planning and fixture planning - Next-Cut	fixturing and machining	modular fixture elements	prismatic	1. fixturing requirements 2. ADs of features 3. minimum material removal 4. geometric interactions	yes	3-axis vertical machining centre	rules; features clustering based on AD of features	component level; set-up forming and sequencing

© 2001 by CRC Press LLC

Fuh et al. 1993.	fixture planning	fixturing	modular fixture elements	prismatic	1. locating datums 2. fixturing constraints 3. tool orientations	yes; CADAM	3-axis vertical machining centre	rules; step-by-step features planning; generate-when-needed strategy	component level; set-up forming and sequencing
Dong, et al. 1991; Dong et al. 1994.	fixture planning	fixturing	modular fixture elements	prismatic	1. ADs of features 2. user-defined fixturing precedence constraints 3. minimum number of orientation changes	yes; ICAD Surface Designer surface models		rules; insertion method of sequencing	component level; set-up sequencing
Yue and Murray, 1994.	fixture planning	fixturing	machining vise	$2\frac{1}{2}$ D prismatic	1. clamping requirements 2. tool ADs	yes; ACIS solid modeller	3-axis machining centre	rules; kinematics forces analysis	component level; set-up forming
Jeng and Gill, 1995.	fixture planning	operation	modular fixture elements	prismatic	1. tool approach direction of features 2. reference and location constraints 3. good manufacturing practices	yes	3-axis vertical machining centre	rules	component level; set-up forming

© 2001 by CRC Press LLC

TABLE 5.3 Set-Up Planning Systems

Authors	Functions/ Name	Viewpoint	Fixturing System	Parts	Criteria	Solid-model Integration	Machining Environment	Reasoning Techniques	Level of Set-up Planning
Hayes and Wright, 1986; Hayes and Wright 1988.	set-up planning - Machinist	fixturing and machining	machining vise	2½ D prismatic	1. geometric features relations 2. machining heuristics 3. stock squaring-up operations	no; feature-based models	3-axis vertical machining centre	rules; features interactions graph and squaring graph merging	component level; set-up forming and sequencing
Chen and LeClair, 1994.	set-up planning - RDS	machining	machining vise	prismatic	1. machining heuristics 2. ADs of features 3. tool commonality	yes	3-axis vertical machining centre	rules; neural network algorithm	component level; set-up forming and sequencing
Ong, et al. 1993; Ong and Nee, 1994a; Ong and Nee, 1994b.	set-up planning - CASP	fixturing and machining	machining vise; modular fixture elements	prismatic	1. geometric relations 2. fixturing requirements 3. tolerances specifications 4. machining heuristics 5. ADs of features	no; feature-based models	3-axis vertical machining centre	rules; fuzzy set theory modeling	component level; set-up forming and sequencing
Zhang et al. 1995.	set-up planning	machining		prismatic	1. machining precedence feature relations 2. ADs of features	no; feature-based models	3-axis vertical machining centre	rules; mathematical optimization algorithm	component level; set-up forming and sequencing
Mei, Zhang and Oldham, 1995; Mei and Zhang, 1992.	set-up planning	machining and fixturing	three-jaws chucks	rotational	1. geometric tolerance requirements 2. workpiece support	no; 11-digit code		neural network	component level; set-up forming and sequencing
Yut and Chang, 1995.	set-up planning	fixturing and machining		2½ D prismatic	1. feasible spindle directions of operations 2. cutting tools 3. roughing and finishing operations	yes; BREP solid models		rules; heuristic algorithm	component level; set-up forming and sequencing
Chu and Gadh, 1996.	set-up planning	fixturing and machining	machining vise	prismatic	1. ADs of features 2. fixturing and referencing requirements 3. machining heuristics	yes		rules	component level; set-up forming and sequencing
Sarma and Wright, 1996.	set-up planning, IMADE	fixturing and machining	machining vise	prismatic	1. access directions of features 2. tool changes 3. stock squaring-up operations 4. machining requirements	yes	3-axis vertical machining centre	graph-theoretic model; DAG graphs	component level; set forming and sequencing

© 2001 by CRC Press LLC

TABLE 5.4 Design Evaluation Systems with Set-up Planning

Authors	Functions/ Name	Viewpoint	Fixturing System	Parts	Criteria	Solid-model Integration	Machining Environment	Reasoning Techniques	Level of Set-up Planning
Das et al., 1994.	design evaluation system	machining		prismatic	1. ADs of features 2. machining precedence constraints	yes; MRSEV solid models	3-axis vertical machining centre	rules; depth-first-branch and bound approach	component level; set-up forming and sequencing
Hayes and Sun, 1995.	design evaluation system	fixturing and machining	machining vise	prismatic	1. tolerances specifications 2. geometric interactions 3. machining precedence	no; feature-based models	3-axis vertical machining centre	rules	component level; set-up forming and sequencing

Features can be modified and resources, such as stock from which the part is made, can be changed. Hayes and Sun (1995) recently reported a rigorous analysis of the tolerance specifications of features for generating redesign suggestions to enhance the cost efficiency. Mäntylä, Opas, and Puhakka [1989] and Das et al. [1994] have also incorporated a methodology for generating redesign suggestions, but are restricted to minor geometrical changes such as fillets and sharp corners. Ong and Nee [1994a] considered a wider range of factors in analyzing the design of the features during set-up planning, providing feedback on the possible changes to the surface finishes, tolerance specifications, fixturability, and geometrical shapes of the features. They performed manufacturability and fixturability analysis of the features while simultaneously formulating a feasible set-up plan.

The aim of *machine batch level set-up planning* is to consider batch and component set-up details to identify the list of set-ups required by a machine batch, and the operations and fixture requirements of each set-up. Machine batch level set-up planning considers the availability of machines and the fixturing constraints for the machine type being considered. This allows problems that relate to specific machines to be considered in detail. It introduces a useful structure in linking and interfacing with downstream activities such as scheduling and capacity planning. Bond and Chang [1988], and Gu and Zhang [1993] have implemented set-up planning at the machine batch set-up level by considering the minimum number of machines that can provide most of the machining operations required by a workpiece. Both these systems performed hierarchical clustering by grouping the features that can be machined by the same machine into clusters, and then examining the approach directions and fixturing requirements to further group the features into smaller clusters, as illustrated in Figure 5.3.

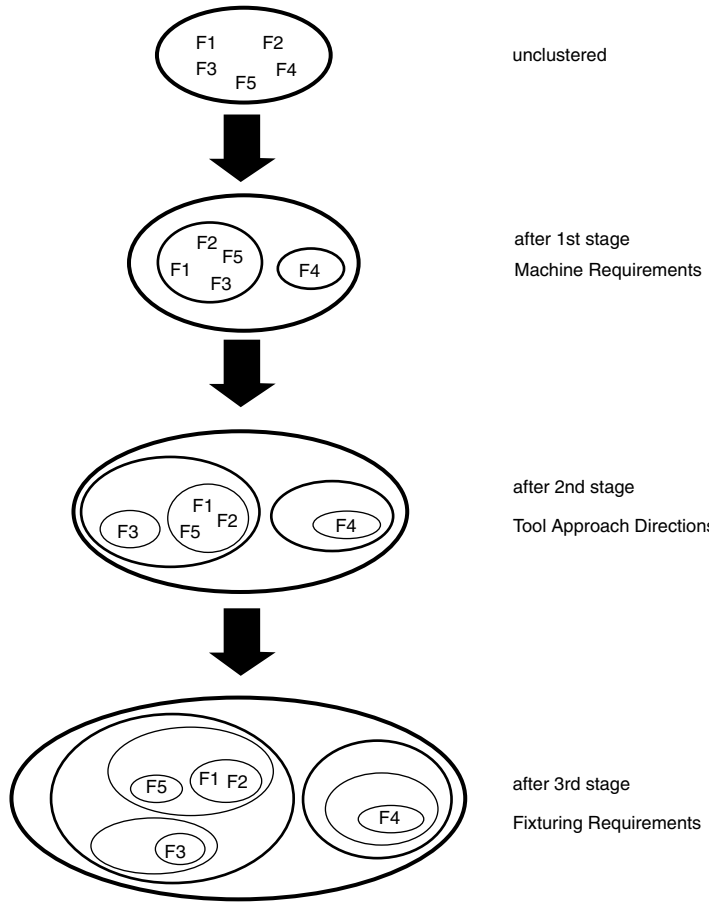


FIGURE 5.3 Hierarchical clustering.

© 2001 by CRC Press LLC

5.3 Two Viewpoints of Set-Up Planning

Generally, set-up planning has been associated with determining the groups of features and/or operations that can be machined together on a particular machine and/or fixture configuration, and the sequencing of these resultant groups. However, a distinction exists in the interpretation of set-up planning by the operations planners and the fixture designers, as illustrated in Figure 5.4. In fixture planning, set-up planning is concerned with the grouping of features and the determination of the orientations of the workpiece for these groups; while in process planning, set-up planning is concerned with the clustering of features into groups and the determination of a machining sequence of these clusters of features/operations. This difference in the concepts of set-up planning has led to a dichotomy in the research and implementation on the automation of the set-up planning process.

The Machining Viewpoint

The objectives of process planning are essentially as follows: (a) select machining processes and tools to generate all the features on a workpiece, (b) select machine tool(s) to perform these required operations, (c) sequence these operations, taking into account features relations, (d) generate set-ups, (e) determine the various requirements for these set-ups, (f) select machining parameters for the operations required, (g) plan the tool paths, and (h) generate the NC part program [Ham and Lu, 1988; Ray and Feenay, 1993; Hetem et al., 1995]. Thus, set-up planning is a part of the generic process planning framework.

Table 5.1 shows the CAPP systems that have included set-up planning, although many listed systems do not associate formulating and sequencing the set-ups as set-up planning. These systems essentially implemented set-up planning from the machining viewpoint. Factors and criteria used are the cutting tools for machining the features, tool cutting paths, dimensional and tolerance requirements, machining directions, etc. The earliest work of implementing set-up planning in a CAPP system was reported by Armstrong, Carey, and de Pennington [1984]. In most of the systems listed in Table 5.1, a set-up is formed by grouping features that have the same approach direction [Joneja and Chang, 1989; Hayes, and Wright, 1986; Chang, 1991], and considering the precedence relationships between the features due to constraints such as spatial and geometrical relationships [Joshi, Vissa, and Chang, 1988; Bond and Chang, 1988; Joneja and Chang, 1989; Warnecke and Muthsam, 1992], dimensional and tolerance specifications [Bell and Young, 1989; Joneja and Chang, 1991; Delbressine, de Groot, and vander Wolf, 1993; Opas, Kanerva, and Mäntylä, 1994; Nordloh, 1994], geometrical accessibility [Delbressine, de Groot, and van der Wolf, 1993; Gu and Zhang, 1993], etc.

An assumption in these systems is that the set-up plans formed will always lead to feasible fixture configurations. The primary objective of these systems is to identify the operations and sequence them, together with the selection of tools and machining parameters. Zhang, Nee, and Ong [1995], Delbressine, de Groot, and van der Wolf [1993], Chen [1993a], Chen and LeClair [1994], Armstrong, Carey, and de Pennington [1984], Gindy and Ratchev [1991], Joshi, Vissa, and Chang [1988], and Bond and Chang [1988] have implemented set-up planning on this basis. Young and Bell [1991] also assumed that the set-up plan formed can be fixtured. This assumption gives these systems an edge over other systems that perform computationally intensive fixture design and planning activities during set-up planning. However, fixturing a set-up is a time-intensive activity [Wiendahl and Fu, 1992]. Thus, this assumption limits the feasibility of the set-up plans and the applicability of these systems, as the work-holding requirements and the availability of fixturing systems are not considered.

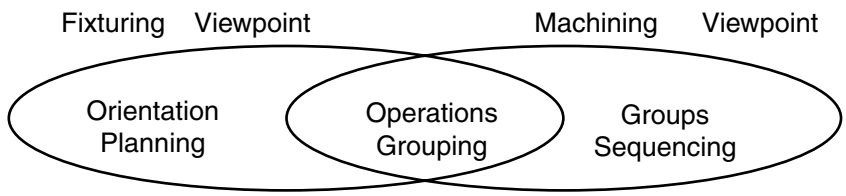


FIGURE 5.4 Set-up planning tasks—machining and fixturing viewpoints.

© 2001 by CRC Press LLC

The Fixturing Viewpoint

The four main objectives in fixture design are to: (a) position a workpiece, in relation to tools, in a desirable orientation, (b) hold the workpiece in the desired position and orientation against tool force, (c) restrict the deflection of the workpiece due to tools and holding forces on the workpiece, and (d) achieve the above three objectives without causing damage to existing machined surfaces [Henriksen, 1973; Wilson and Holt, 1962; Boyes, 1989].

From the fixturing viewpoint, set-up planning groups features to be made, and determines the orientations to make these features in certain fixturing configurations. It includes the configuration, as well as the location of the fixturing components. Figure 5.5 shows the activities and constraints of set-up planning from the fixturing viewpoint. A set-up is thus a fixturing configuration for processing features using tools and fixturing elements. Several fixture planning systems have incorporated set-up planning

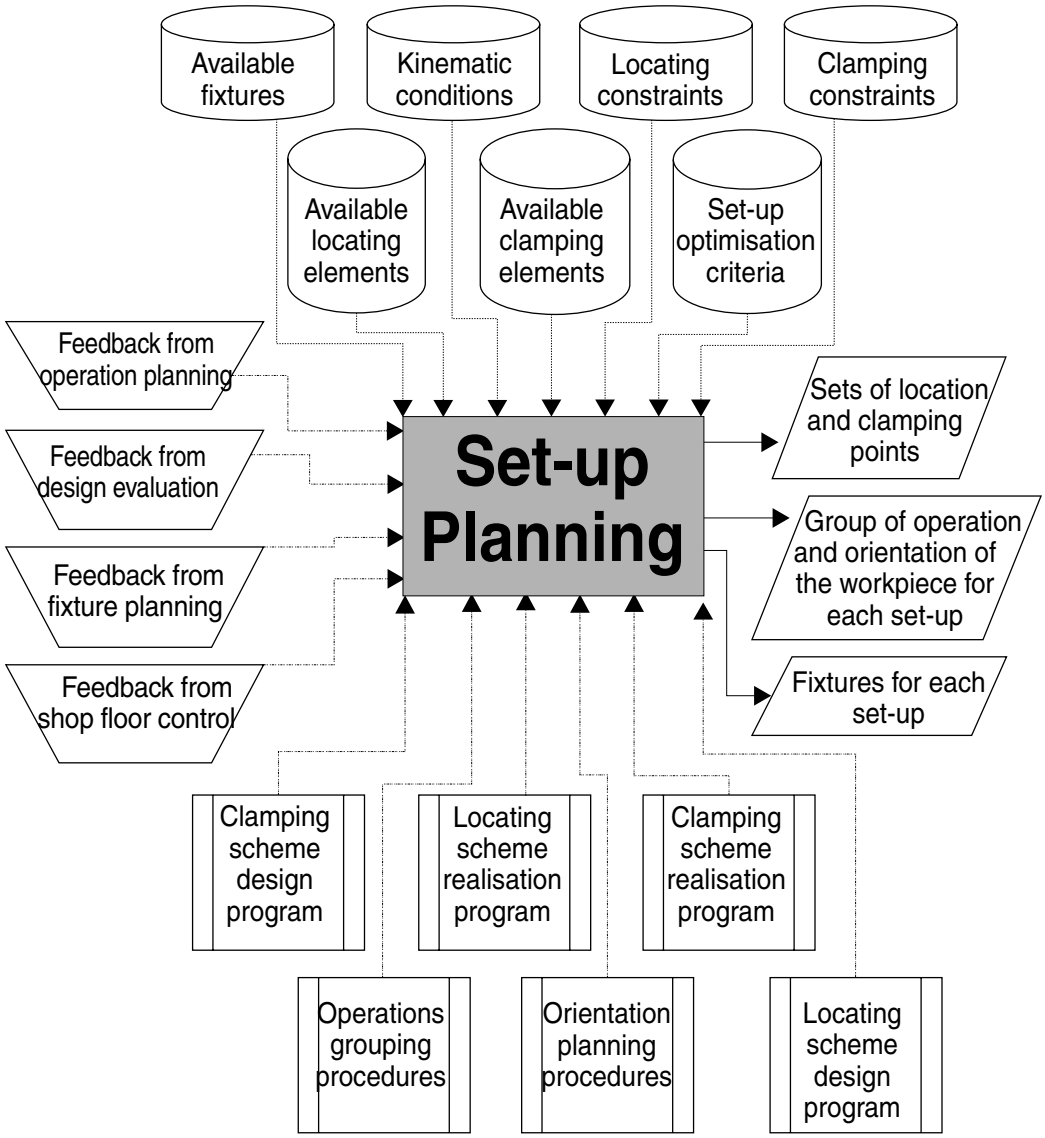


FIGURE 5.5 Fixturing viewpoint of set-up planning.

© 2001 by CRC Press LLC

in their implementation, while other CAFP systems assume the set-up plans to be available and design fixture configurations according to these set-up plans [Ong et al., 1993; Ong and Nee, 1994a; Chang, 1992]. Table 5.2 lists the CAFP systems that have included set-up planning in their implementation. The fixturing viewpoint set-up planning approach uses fixturing criteria and work-holding requirements of the workpieces for generating set-ups.

A critical analysis of the fixturing viewpoint systems listed in Table 5.2 shows that the types of work-holding devices have a great impact on the procedures and reasoning needed to obtain the set-up plans. Chan and Voelcker [1986], Englert and Wright [1986], Hayes and Wright [1986; 1988], Hayes [1987] Gu and Zhang [1993], and Yue and Murray [1994] have reported systems that perform set-up planning as a part of fixture planning and design based on the availability of machine vices. In the system reported by Chan and Voelcker [1986], fixturing requirements of a workpiece using machining vices are interactively specified by the users. The machinist system reported by Hayes [1990], on the other hand, automatically proposes a squaring graph that outlines all the methods for converting a raw material into a square shape, based on machining vices. She also defined an interaction graph that identifies possible feature interactions and resultant features sequences, to process these features from pre-defined rules acquired from machinists. By finding the commonality between the squaring and interaction graphs, set-up and feature sequences can be roughly determined. The use of squaring-up plans has been adopted by many other researchers such as Chu and Gadh [1996], and Sarma and Wright [1996]. A rigorous clamping strategy for formulating set-up plans was recently reported by Yue and Murray [1994] for 2½D prismatic workpieces. This methodology selects a clamping strategy based on machining vices by evaluating the areas of the faces to constrain the workpiece against machining and clamping forces, possible deformation of the workpiece, clearance between the features and the machining vice, workpiece overhang, and approach directions of features [Corney et al., 1992] during set-up planning.

Set-up planning using modular fixture elements is a more complex problem [Ong and Nee, 1994b; Yue and Murray, 1994] than had been attempted by Boerma and Kals [1988, 1989], Ferreira and Liu [1988], Sakurai and Gossard [1991], Lee [1991], Kambhampati et al. [1993], Fuh, Chang, and Melkanoff [1993], Ong and Nee [1994a], and Dong, DeVries, and Wozny [1994; 1991]. The main criteria in these systems are the supporting and locating requirements, in addition to the clamping requirements of the workpieces, the datum referencing requirements, and the machining directions of the features. In these systems, set-up planning begins by selecting the locating and datum surfaces for referencing the workpieces in the fixture configurations with respect to the cutting tools. This is different compared to machining vices, in which the first stage is usually the selection of a suitable vice.

Three trends can be observed among systems that have implemented set-up planning as listed in Table 5.2. One group determines the fixturing feasibility of every set-up so that the resultant set-up plan is feasible. Joneja and Chang [1991] have incorporated the test for fixturing feasibility into their clustering and refinement procedure for determining the set-up plan for a workpiece. In their system, the fixturing feasibility of every set-up is determined based on a set of available machining vices. A fixturing module evaluates the fixturing feasibility of a set-up and returns a list of features that could be machined. Set-ups that cannot be fixtured will be refined to form more clusters and further evaluated. This methodology is similar to the work reported by Lee, Cutkosky, and Kambhampati [1991], Kambhampati et al., [1993], and Brown, Cutkosky, and Tenebaum [1991] on Next-Cut, in which set-up planning is performed simultaneously with fixture planning to incrementally formulate a set-up plan for a workpiece. In Next-Cut, a set-up graph is formed based on the approach directions of features, by merging the feasible tool paths with the process graphs of features. A fixture agent checks whether a fixturing arrangement can be generated for each candidate set-up in the set-up graph.

Another group adopts a different strategy when generating the set-up plan from the fixturing viewpoint. Fuh, Chang, and Melkanoff [1993], and Sakurai and Gossard [1991] implemented set-up planning together with fixture planning. Fuh considered one feature at a time from a set of features required to be machined. In their system, a new set-up is formed when a feature cannot be included in an existing set-up due to constraints such as machining directions and datum referencing requirements. This would correspond to a new fixturing configuration. Sakurai and Gossard estimated the clamping forces and the possible deformation of workpieces, and checked the locating accuracy and interference to determine the goodness of a set-up. In both systems, the fixturing constraints are automatically generated. Dong,

© 2001 by CRC Press LLC

Gilman, and Wozny [1994], on the other hand, implemented set-up planning using the same method as Fuh and Sakurai and Gossard, but the fixturing constraints are interactively specified by the users and transformed into dynamic constraints during set-up planning.

A third approach is to consider the fixturing requirements and constraints of the features, and the work-holding systems that are available during set-up planning, but without determining a fixture plan for the set-up plan that is formed. This approach has the advantage of providing the flexibility on the selection of the fixture elements for each set-up. Ong and Nee [1994a] transformed the fixturing requirements of workpiece features into fuzzy fixturing features relations during set-up planning. The availability of the fixture elements is also considered using a define-by-fixture-features methodology and a feature-to-feature mapping procedure between the fixturing features and the fixture-features on the modular fixture elements.

5.4 Factors and Constraints in Set-Up Planning

The objectives of set-up planning are to (a) identify groups of features that can be machined in a single set-up, (b) determine a desirable workpiece orientation for each set-up, (c) determine an appropriate fixturing method for each set-up, and (d) determine set-ups order for machining. A set of features to be generated on a workpiece has explicit and implicit relations with each other and the features already present. These relations are formed due to factors and constraints such as (a) the machining directions of features, (b) the geometrical relationships of features, (c) the datum and referencing requirements of features, (d) processes, tools, and machines for workpiece manufacturing, (e) intermediate states of the workpiece, (f) the fixturing requirements of the features, and (g) good manufacturing practices, as shown in Figure 5.6. Regardless of the nature of these relations, they have significant effects on the set-ups formulation for producing a workpiece and their order.

Approach Directions of Features

In most current set-up planning methodologies, a set-up is formed by grouping features that have the same approach direction (AD). A number of terms have been used interchangeably with the approach directions of features, such as tool access directions of features [Mayer, Su, and Keen, 1992], spindle axis directions

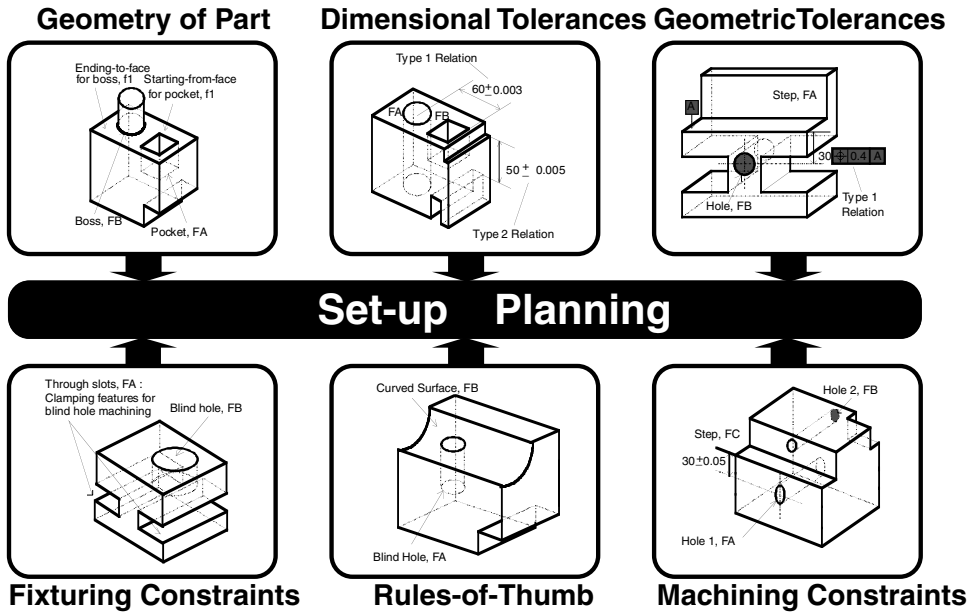


FIGURE 5.6 Factors in set-up planning.

© 2001 by CRC Press LLC

[Warnecke and Muthsam, 1992; Joneja and Chang, 1991; Joshi, Vissa and Chang, 1988], tools orientations [Fuh, Chang, and Melkanoff, 1993], machining directions [Mäntylä, Opas, and Puhakka, 1988; Opas, Kanerra, and Mantyla, 1994; Jeng and Gill, 1995], and approach faces [Westhoven et al., 1992; Chen, 1993a]. This is the first criterion in many reported set-up planning systems. For a three-axis machining center, an approach direction is equivalent to a set-up direction. The production cost of a machined component is closely related to the number of set-ups needed; hence, the approach direction is an important consideration in set-up planning.

In the system reported by Chu and Gadh [1996], planar faces are treated as features with an infinite number of approach directions, while other researchers have generally treated planar faces as non-features, i.e., not a form feature. Corney et al. [1992] reported an algorithm for determining the approach directions of features on 2½D prismatic parts. Other systems have generally assumed the approach direction of a form feature to be an unobstructed path that a tool can take to access that feature on a part. These systems essentially cluster features together based on their approach directions and sequence the resultant set-ups that are formed, combining and eliminating set-ups that are redundant. A set-up is redundant if all its features belong to other set-ups. It is possible to have a feature in more than one set-up during the initial set-up planning stage, since a feature can have more than one approach direction. Such systems need to detect whether a feature is in other clusters [Chang, 1990]. This process requires the comparison of all the features in every cluster.

Gindy and Ratchev [1991], Young and Bell [1991], Bell and Young, [1989], Mäntylä and Opas [1988], Joneja and Chang [1989]; Chang [1991], Warnecke and Muthsam [1992]; Muthsam [1990], Chu and Gadh [1996], and Sarma and Wright [1996] have also implemented set-up planning using the approach directions of features as a criterion for clustering the features to form an initial, loosely constrained, set-up plan. In these systems, it is assumed that a feature can be machined with equal ease from any of its approach directions. However, it is preferred to machine a feature in the approach direction with the smallest depth of cut. Ong and Nee [1994a] have formulated an algorithm for calculating a preferential index for each approach direction of a feature.

Geometrical Relationships

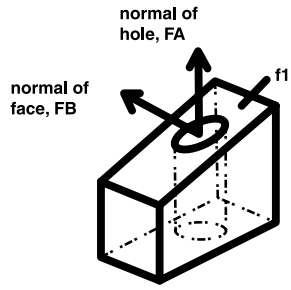
The geometrical shape of the features present on the workpieces would stipulate some relationships with the features to be machined. Examples of geometrical relationships are the opening-to-face and the starting-from-face relationships between the features and the faces on the workpieces. These geometrical relationships between neighboring features are inherent in the design of a workpiece and the physical attributes of the machined surfaces. They cannot be violated since they physically limit the machining of surfaces or shape elements. Geometrical interactions of features are the main planning criterion for set-up planning in systems reported by Hayes and Wright [1988], and Chen [1993c].

Design Specifications

Dimensions and tolerances [ANSI, 1982] of a part are specified by a designer with regard to the functions of the part. The process planners and fixture designers have to follow the dimensional and geometrical tolerances of a part closely during the planning process. Machining difficulty depends on the specified tolerances, and closely toleranced features have to be machined in one set-up. The systems reported by Boerma and Kals [1988, 1989]; Boerma [1990], and Delbressine, de Groot, and van der Wolf [1993] considered the tolerance specifications of a part as the main criterion in set-up planning. Boerma and Kals proposed a tolerance conversion scheme [1990] to convert each tolerance specification to a tolerance factor. In each case, features with the most critical tolerances will be machined last. Delbressine represented these tolerance factors using a tolerance graph.

The dimensional and geometric tolerances of the features on a part have also been used for set-up planning in the systems reported by Zhang, Nee, and Ong [1995], Ong and Nee [1994a], Joneja and Chang [1989], Chu and Gadh [1996], and Dong, Gilman, and Wozny [1994]. In these systems, tolerances are only one of the factors in set-up planning. Zhang, Nee, and Ong [1995] and Dong, Gilman, and Wozny [1994]

© 2001 by CRC Press LLC



IF f1 is *starting-from-face* for FA
AND f1.nor_dir is not equal to FA.AD
AND f1 is a *comprising face* of FB
THEN FB _ FA, T = HR, DG = 0.9
where _ denotes dependent, AD denotes approach direction, T denotes type of relation, HR denotes Heuristic Relation, and DG denotes dependency grade.

FIGURE 5.7 A rule-of-thumb.

considered the datum referencing requirements of the tolerances during set-up planning, while Joneja and Chang considered essentially the positional tolerances of the features. In the system by Chu and Gadh, datum referencing requirements are treated as reference edge interactions which should be avoided in a set-up plan. In the system reported by Ong and Nee [1994b], the tolerance conversion scheme proposed by Boerma [1990] is used. However, this system fuzzifies the tolerance factors that are being derived and compares them with other fuzzified factors such as the geometrical relations on a common basis.

Machining Requirements

Prior executions of operations such as squaring up to prepare the workpiece for subsequent machining may require additional set-ups. In practice, rules-of-thumb and good manufacturing practices have been established to ensure an accurate, efficient, and economic manufacture of workpieces [Trucks, 1987; Dallas, 1976]. Many machining handbooks provide such machining guidelines [Boothroyd, Dewhurst, and Knight, 1994; Bralla, 1986; MetCut, 1972]. Figure 5.7 shows such an example. Other manufacturing rules are, for example, a face should not be followed by a hole which direction of the central axis is not parallel to the normal of the face, and a hole should not be followed by a face which normal vector is parallel to the direction of the central axis of the hole. Many systems shown in Tables 5.1 to 5.3 have coded rules-of-thumb into production rules for determining the machining precedence relations between features.

Fixturing Requirements

A set-up plan must be designed in such a way a part can be accurately held in a work-holding device. The plan should consider the stability of a part in the fixtures and ensure that there is minimum deformation of the part under machining and clamping forces.

5.5 Features Interactions in Set-Up Planning

Feature interaction was first suggested by Hayes and Wright [1986] to refer to the relationships between features that would affect the process planning function. These interactions essentially refer to the geometric relationships between features that cause complications in the process planning task. This concept of feature interactions has been adopted by many researchers such as Joneja and Chang [1989]; Chang [1990, 1991], Zhang, Nee, and Ong [1995], Ong and Nee [1994a], Chu and Gadh [1996], Sarma and Wright [1996], Chen et al. [1993b], and Chen and LeClair [1994] for set-up planning. The idea of features interactions has been used in the systems reported by Westhoven et al. [1992], Weill, Spur, and Eversheim [1982], Nevrinceanu and Donath [1987a], Nevrinceanu, Morellas, and Donath [1993], and Züst and Taiber [1990] for determining the machining sequence of features.

In the machinist system reported by Hayes and Wright [1988] and Hayes [1990], the set-up plan is refined based on the avoidance of negative feature interactions and the overlapping of positive feature interactions.

© 2001 by CRC Press LLC

Feature-specific rules are built into the system to identify and avoid negative feature interactions by re-ordering a sequence of set-ups, putting features into different set-ups, or generating new set-ups to contain these features. In the system by Jung and Lee [1994], only geometric features interactions are considered. These geometric interactions are represented by a feature interaction graph. Hwang and Miller [1995] used a hybrid blackboard model that uses mixed-type reasoning to handle topological and tolerance interactions between features. Joneja and Chang [1989], Chang [1990, 1991] considered these interactions, together with the location tolerances and reference specifications of a part, through the use of precedence relations between features. The various feature interactions, due to the geometric orientations of the features, only serve to refine the plans, which are initially formulated based on the approach directions of these features. Chen et al. [1993b] and Chen and Leclair [1994] used an episodal association memory (EAM) technique to organize geometric feature interactions. In the object-oriented rule-based system reported by Ong and Nee (1994a), features relations are the main planning objects. They include the constraints caused by factors such as the dimensional and tolerance specification of the features, the machining and fixturing requirements of the features, and manufacturing heuristics, besides the geometrical interactions.

5.6 Artificial Intelligence and Set-Up Planning

Many intelligent computing techniques have been developed and applied over the last decade to model and automate manufacturing planning functions [Madey, Weinroth, and Shah, 1994], as shown in Figure 5.8. Some of these techniques include neural networks, fuzzy systems, genetic algorithms, rule-based production systems, and a wide spectrum of techniques lumped under the heading of artificial intelligence [Badiru, 1992; Dagli, 1994]. Advanced modeling and reasoning techniques such as case-based reasoning, constraint-based reasoning, fuzzy logic, and fuzzy associative memory, have been applied to the modelling

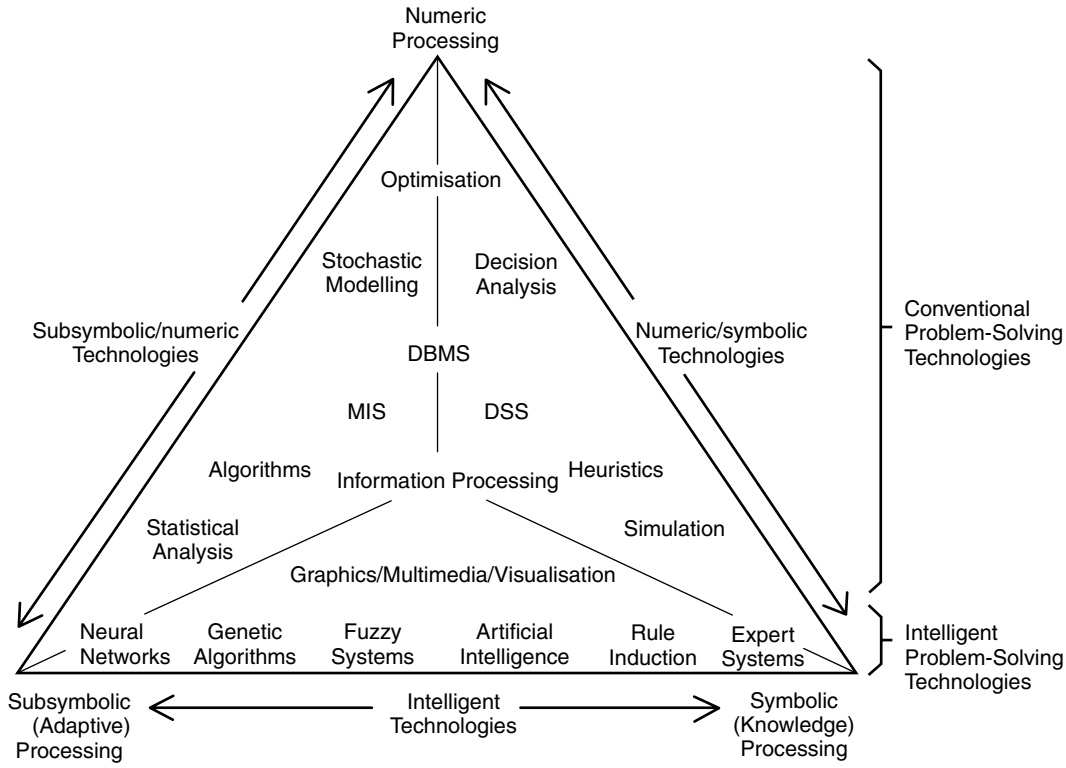


FIGURE 5.8 Problem-solving methods for manufacturing. [Adapted from Madey, G.R., Weinroth, J., and Shah, V., 1994, Hybrid intelligent systems, tools for decision making in intelligent manufacturing, in *Artificial Neural Networks for Intelligent Manufacturing*, Dagli, C.H., Ed., 67–90, Chapman & Hall, London.]

© 2001 by CRC Press LLC

of human reasoning processes in formulating manufacturing plans, and the development of computer-aided manufacturing planning systems [Dym and Levitt, 1991; Wiendahl and Scholtissek, 1994; Eskicioglu, 1992].

Set-up planning is a mixture of complex and inter-related tasks as shown in Figure 5.5. A number of the activities in set-up planning can be solved by applying algorithms and technological constraints, such as the evaluations of the clamping and cutting forces [Lee, Cutkosky, and Kambhampati, 1991; Fuh, Chang, and Melkanoff, 1993; Ferreira and Liu, 1988; Yue and Murray, 1994], and the locating accuracy of the workpiece [Sakurai, 1992; Ferreira and Liu, 1988]. In other cases, certain sub-tasks can be formulated analytically, but the number of alternative solutions can be very large. Finally, other problems, such as the determination of precedence features relations cannot be solved analytically, and their determination depends upon the knowledge available within the domain. It therefore follows that both analytical methods and knowledge-based systems are suitable and required for performing the various procedures in set-up planning. Typical examples of the analytical approach are the optimization of the locating accuracy and fixturing points using optimization algorithms and the evaluation and formulation of set-up plans using the fuzzy set theory [Ong and Nee, 1994a]. Knowledge-based techniques, on the other hand, are particularly suitable for the determination of feature relations and interactions, and the selection, grouping, and sequencing of operations. Many of the systems listed in Table 5.1 are rule-based and/or knowledge-based systems. Production rules are employed to encode the machining requirements, fixturing constraints, and geometrical, dimensional, and tolerance specifications to determine the precedence relations and features interactions between features [Hayes and Wright, 1986; Chang, 1991; Mäntylä, Opas, and Puhakka, 1988; Joneja and Chang, 1989; Young and Bell, 1991].

Artificial intelligence techniques such as EAM [Westhoven et al., 1992], constraint propagation network [Nevrinceanu, Morellas, and Donath, 1993], and matrices representations [Weill, Spur, and Eversheim, 1982; Züst and Taiber, 1990] have been used in operations sequencing systems for the knowledge representation and processing of feature interactions and relations.

Chen [1993a] and Chen and LeClair [1994] adopted the EAM technique by Westhoven, which is an associative memory that integrates dynamic memory organization and neural computing technologies for organizing geometric features relations. A self-learning neural-network algorithm is used together with EAM to formulate set-up plans. In the system reported by Englert and Wright [1986], features relations are simplified into a binary table of cuts to orientate and machine the features. Graph representation of these features relations was used in the more recently reported systems to determine the sequence of machining the set-ups. Hayes, Desa, and Wright [1989] used an interaction graph together with a squaring graph to formulate set-up plans, while Lee, Cutkosky, and Kambhampati [1991]; Kambhampati et al. [1993]; Brown, Cutkosky, and Tenebaum, [1991] used a set-up graph, which is formed based on the features relations, with a process graph. The features relations are transformed into a tolerance graph and a precedence graph in the IDM system reported by Delbressine, de Groot, and van der Wolf [1993]. A hybrid blackboard model using a mixed-type reasoning technique was reported by Hwang and Miller [1995]. This hybrid blackboard model handles features interactions by combining the forward chaining for feature sequencing with the backward chaining for the construction of process plans. All these systems use some conflict-resolving procedures to resolve the conflicts between features relations. However, the issue of concurrently considering all the features relations in the set-up planning process, as performed by human machinists, is still not solved in these systems. Ong and Nee [1994a] addressed this issue in their CASP system that simultaneously considers all the features relations during set-up planning by representing every feature relation as a fuzzy relation, and transforming all the features relations into a fuzzy matrix to derive the set-up plan. In this system, all the factors are concurrently considered to formulate a set-up plan. The use of fuzzy sets and fuzzy relations also provides a means of representing the degrees of importance of the features relations, since in real-life, not all factors will affect the machining process in the same degree.

5.7 Open Research Issues

The research into set-up planning has so far established scientific principles, uncovered useful methodologies for set-up formulation and sequencing, and demonstrated useful results. However, several generic issues would require continuing research. Specific research issues in set-up automation include (a) set-up

© 2001 by CRC Press LLC

validity and optimization, (b) integration and interfacing with design activities, and (c) integration and interfacing with shop floor control activities.

Set-Up Validity and Optimization

The essential idea of set-up validity is that the set-up planning system should be able to compute whether the position and orientation of a workpiece are completely fixed by the specified set-up operations, whether the workpiece is stable under gravity and small disturbances, and whether the forces generated by machining and table motion will exceed the direct and induced clamping forces, besides the selection and sequencing of the operations for generating the features and the determination of an orientation for each set-up that is formed. One of the difficulties of set-up planning is that the goals of the various sub-tasks can be contradictory. Hence, the overall planning problem cannot be decomposed into independent sub-problems. The global goal of set-up planning is to reduce the number of approach directions to a minimum; even if a machining center is capable of machining a workpiece from several approach directions without a second set-up, since fixture design can be simplified if fewer approach directions are used. In contrast, the local goal of tool selection for a single feature is to select the best possible process for that feature. If the machining features of a complex workpiece are all treated independently in terms of tool and approach direction, an inefficient overall process plan may result. To achieve a better plan, it is necessary to consider the interactions of all the features as a whole, in addition to each feature separately [Willis et al., 1989; Ong and Nee, 1994a]. Many existing systems do not consider the set-up planning process as a multi-objective problem as stated above. These systems used criteria such as minimizing the number of set-ups required and maximizing the number of critical tolerances attained. Most reported set-up planning methodologies produce acceptable, but not optimal, set-up plans. Hence, multi-objective optimal set-up planning is still an open research issue.

Set-Up Planning and Product Design

Design evaluation of a part has to be performed concurrently with manufacturing planning functions to achieve full integration. However, in most of the present systems, parts are seldom evaluated during the design stage, but more towards prototyping and manufacturing. Set-up planning links up the various activities from design to production (Figure 5.9). Automating set-up planning is therefore crucial for integrating the activities in manufacturing a part. As mentioned in Section 5.2, a few reported set-up planning systems have incorporated redesign suggestions. However, these systems are presently restricted to providing redesign suggestions only at the end of the set-up planning process. The task of concurrently performing set-up planning and design refinement is still an area that has not been explored. The ability to closely knit the set-up planning activities with the design activities would be a significant step towards the achievement of a concurrent engineering environment.

Set-Up Planning and Shop Floor Control

Shop floor control, a function at the downstream level in a manufacturing system as shown in Figure 5.9, comprises scheduling, dispatching, monitoring, and diagnostics. It receives process and capacity plans from the design and planning levels. According to these plans, a workplan for a specific production period is generated. Jobs and related auxiliary tasks are scheduled and released. Auxiliary tasks such as tool preparations are usually carried out preceding the actual execution of the job. A job can be defined as a set of coherent operations allocated to a workstation, e.g., the machining of a single workpiece in a single set-up on a machining center. The execution of the jobs is monitored and discrepancies between the schedules and the actual progress are analyzed. Diagnostic results are fed back to the scheduler. A tight integration between CAPP and scheduling is essential for the realization of concurrence in process planning and production scheduling. In addition, in a manufacturing environment there are usually several resources, e.g., a range of CNC machines, which are capable of producing a workpiece. Capacity planning is employed to schedule various workpieces on the machines to maximize resource utilization,

© 2001 by CRC Press LLC

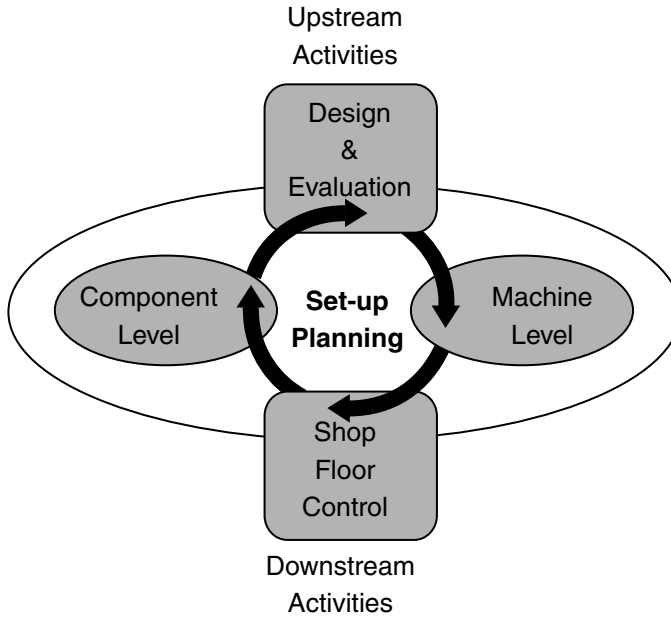


FIGURE 5.9 Set-up planning—the link between design and production.

and, at the same time, achieve maximum production. Feedback from the shop floor level is therefore essential for the formulation of practical and feasible set-up plans that take into consideration the actual resources available. To develop set-up plans for a range of machining resources will be a significant step in linking process planning and fixture design with shop floor control.

5.8 Summary

The importance of automating the set-up planning process in the manufacturing environment has been discussed and examined from two different perspectives, namely, the machining and the fixturing viewpoints. An automated set-up planning system that could simultaneously consider the constraints from both perspectives would be able to formulate practical and feasible set-up plans, and further enhance the existing CAPP and CAFP systems. In addition, this paper analyzes set-up planning at the machine batch set-up level and the component set-up level, to determine the relationships of set-up planning at the design stage of a product and the shop floor activities when producing this product. Set-up planning is a crucial link for integrating and interfacing the various activities, from design to manufacturing, in achieving concurrent engineering. The paper concludes with a discussion of a few open research issues in set-up planning that require further research.

References

Alting, L. and Zhang, H.-C., 1989, Computer-aided process planning: the state-of-the-art survey, *Int. J. Prod. Res.* 27, 553–585.

ANSI Y14.5M, 1982, *Dimensioning and Tolerancing*, ASME.

Anderson, D.C. and Chang, T.C., 1990, Geometric reasoning in feature-based design and process planning, *Comput. & Graphics J.,* 14, 225–235.

Armstrong, G.T., Carey, C.G., and de Pennington, A., 1984, Numerical code generation from a geometric modelling system, in *Solid Modelling by Comput.,* Plenum Press, New York.

© 2001 by CRC Press LLC

Badiru, A.B., 1992, *Expert Systems Applications in Engineering and Manufacturing*, Prentice-Hall, Englewood Cliffs, NJ.

Bell, R. and Young, R.I.M., 1989, Machine planning: its role in the generation of manufacturing code from solid model descriptions, *Int. J. Prod. Res.,* 27, 847–867.

Boerma, J.R., 1990, The design of fixtures for prismatic parts, Ph.D. thesis, University of Twente, The Netherlands.

Boerma, J.R. and Kals, H.J.J., 1988, FIXES, A system for automatic selection of set-ups and design of fixtures, *Annals of the CIRP,* 37, 443–446.

Boerma, J.R. and Kals, H.J.J., 1989, Fixture design with FIXES: the automatic selection of positioning, clamping and support features for prismatic parts, *Annals of the CIRP,* 38, 399–402.

Bond, A.H. and Chang, K.J., 1988, Feature-based process planning for machined parts, *Proceedings of Comput. in Eng. Conf*., ASME, 1, 571–576.

Boothroyd, G., Dewhurst, P., and Knight, W., 1994, *Product Design for Manufacture and Assembly,* Marcel Dekker, NY.

Boyes, W.E., 1989, *Handbook of Jig and Fixture Design,* 2nd ed., SME.

Bralla, J.G., 1986, *Handbook of Product Design for Manufacturing*, McGraw-Hill, New York.

Bronsvoort, W.F. and Jansen, F.W., 1993, Feature modelling and conversion — key concept to concurrent engineering, *Comput. Ind.,* 21, 61–86.

Brown, D.R., Cutkosky, M.R., Tenebaum, J.M., 1991, Next-cut: a second generation framework for concurrent engineering, in *Computer-Aided Co-operative Product Development,* D. Sriram, Ed., Springer-Verlag, New York.

Bullinger, H.-J., Warnecke, H.J., and Lentes, H.-P., 1986, Towards the factory of the future, *Int. J. Prod. Res.,* 24, 697–741.

Chan, S.C. and Voelcker, H.B., 1986, An introduction to MPL— a new machining process/programming language, *Proc. IEEE Int. Conf. Robotics and Autom.,* 1, 333–344.

Chang, C.H., 1992, Computer-assisted fixture planning for machining processes—problems and approaches, *Manuf. Rev.,* ASME, 5, 15–28.

Chang, T.C., 1990, *Expert Process Planning for Manufacturing,* Addison-Wesley, Reading, MA.

Chang, T.C., 1991, The quick turnaround cell—an integrated manufacturing cell with process planning capability, in *Manufacturing Cells: Control, Programming and Integration.* D.F. Williams, and P. Rogers, Eds., 118–142, Butterworth-Heiermann Ltd., U.K.

Chen, C.L.P., 1993, Set-up generation and feature sequencing using unsupervised learning algorithm, *Proc. 1993 NSF Design and Manuf. Syst. Conf.,* SME, 2, 981–986.

Chen, C.L.P., 1993, Set-up generation and feature sequencing using an unsupervised learning algorithm, in *Neural Networks in Design and Manufacturing,* J. Wang, and Y. Takefuji, Eds., 135–162, World Scientific.

Chen, C.L.P., Westhoven, T.E., Pao, Y.-H., and LeClair, S.R., 1993, Episodal associative memory approach for sequencing interactive features, *Proc. 1993 NSF Design and Manuf. Syst. Conf.,* SME, 2, 987–991.

Chen, C.L.P. and LeClair, S.R., 1994, Integration of design and manufacturing: solving set-up generation and feature sequencing using an unsupervised-learning approach, *J. Comput.-Aided Design,* 26, 59–75.

Chu, C.C.P. and Gadh, R., 1996, Feature-based approach for set-up minimisation of process design from product design, *J. Comput.-Aided Design,* 28(5), 321–332.

Corney, J., Clark, D.E.R., Murray, J.L., and Yue, Y., 1992, Automatic Classification of 2½D Components, *Concurrent Eng.,* PED-Vol. 59, ASME, 85–99.

Dagli, C.H., 1994, *Artificial Neural Networks for Intelligent Manufacturing,* CRC Press, Boca Raton, FL.

Dallas, D.B., 1976, *Tool and Manufacturing Engineers Handbook,* 3rd ed., McGraw-Hill, New York.

Das, D., Gupta, S.K., and Nau, D.S., 1994, Reducing set-up cost by automated generation of redesign suggestions, *Proc. Comput. Eng. Conf.,* ASME, 159–170.

© 2001 by CRC Press LLC

Delbressine, F.L.M., de Groot, R., and van der Wolf, A.C.H., 1993, On the automatic generations of set-ups given a feature-based representation, *Annals of the CIRP,* 42, 527–530.

Dong, X., DeVries, W.R., and Wozny, M.J., 1991, Feature-based reasoning in fixture designs, *Annals of the CIRP,* 40, 111–114.

Dong, X., Gilman, C.R., and Wozny, M.J., 1994, Feature-based fixture design and set-up planning, in *Artificial Intelligence in Optimal Design and Manufacturing,* Z. Dong, Ed., 5–31, Prentice-Hall, Englewood Cliffs, NJ.

Dym, C.L. and Levitt, R.E., 1991, *Knowledge-Based Systems in Engineering,* McGraw-Hill, New York.

ElMaraghy, H.A., 1993, Evolution and future perspectives of CAPP, *Annals of the CIRP,* 42, 739–751.

Englert, P.J. and Wright, P.K., 1986, Applications of artificial intelligence and the design of fixtures for automated manufacturing, *Proc. IEEE Int. Conf. on Robotics and Autom.,* 1, 345–351.

Englert, P.J. and Wright, P.K., 1988, Principles for part set-up and workholding in automated manufacturing, *J. Manuf. Syst.,* 7, 147–161.

Eskicioglu, H., 1992, The use of expert system building tools in process planning, *Eng. Appl. Artif. Intelligence,* 5, 33–42.

Ferreira, P.M. and Liu, C.R., 1988, Generation of workpiece orientations for machining using a rule-based system, *Robotics & Comput.-Integrated Manuf. J.*, 4, 545–555.

Fuh, J.Y.H., Chang, C.-H., and Melkanoff, M.A., 1993, An integrated fixture planning and analysis system for machining processes, *Robotics* & *Comput.-Integrated Manuf. J.,* 10, 339–353.

Gindy, N.N.Z. and Ratchev, T.M., 1991, Product and machine tools data models for computer aided process planning systems, in *Comput. Appl. Prod. and Eng.: Integration Aspects*, G. Doumeingts, J. Browne, and M. Tomljanovich, Eds., 527–534, Elsevier Science, New York.

Gu, P. and Zhang, Y., 1993, Operation sequencing in an automated process planning system, *J. Intelligent Manuf.,* 4, 219–232.

Ham, I. and Lu, S.C.-Y., 1988, Computer-aided process planning: the present and the future, *Annals of the CIRP,* 37, 591–601.

Hargrove, S.K. and Kusiak, A., 1994, Computer-aided fixture design: a review, *Int. J. Prod. Res.,* 32, 733–753.

Hayes, C.C., 1987, *Using Goal Interactions to Guide Planning: The Program Model,* Technical Report, CMU-RI-TR-87–10, Carnegie Mellon University, Pittsburgh.

Hayes, C.C., 1990, Machining Planning: a model of an expert level planning process, Ph.D. dissertation, Carnegie-Mellon University, Pittsburgh.

Hayes, C.C. and Wright, P.K., 1986, Automated planning in the machining domain, *Knowledge Based Expert Syst. Manuf.,* PED-Vol. 24, 221–232.

Hayes, C.C. and Wright, P.K., 1988, Automating process planning: using feature interactions to guide search, *J. Manuf. Syst.,* 8, 1–14.

Hayes, C.C. and Sun, H.C., 1995, Using a manufacturing constraint network to identify cost-critical areas of designs, *Artificial Intelligence for Eng. Design, Anal. and Manuf. J.*, 9, 73–87.

Hayes, C.C., Desa, S., and Wright, P.K., 1989, Using process planning knowledge to make design suggestions concurrently, *Concurrent Product and Process Design,* Winter Annual Meeting of ASME, PED-Vol. 36, 87–92.

Henriksen, E.K., 1973, *Jig and Fixture Design Manual*, Industrial Press.

Hetem, V., Carr, K., Lucenti, M., Ruiz, O., Zhu, X., Ferreira, P.M., and Lu, S.C.-Y, 1995, Specification for a process planning enabling platform, *J. Syst. Eng.,* 5, 48–59.

Hwang, J.S. and Miller, W.A., 1995, Hybrid blackboard model for feature interactions in process planning, *Comput. Industrial Eng. J.,* 29(1–4), 613–617.

Jeng, Y.C. and Gill, K.F., 1995, An intelligent CAD system for automating fixture design, in *Advances in Manuf. Technol.,* IX, Proceedings of the 11th National Conference on Manufacturing Research, D. Stockton, and C. Wainwright, Eds., 194–198, Taylor & Francis, London.

© 2001 by CRC Press LLC

Joneja, A. and Chang, T.C., 1989, A generalised framework for automatic planning of fixture configuration, *Advances in Manuf. Syst. Eng.,* ASME Winter Annual Meeting, PED-Vol. 37, 17–28.

Joneja, J. and Chang, T.C., 1991, Search anatomy in feature-based automated process planning, *J. Design and Manuf.,* 1, 7–15.

Joshi, S., Vissa, N., and Chang, T.C., 1988, Expert process planning system with a solid model interface, *Expert Systems: Design and Management of Manufacturing Systems,* A. Kusiak, Ed., 111–136, Taylor & Francis, London.

Jung, M.Y. and Lee, K.H., 1994, Conceptual framework for automated process planning, *Comput. Ind. Eng. J.*, 27(1–4), 115–118.

Kambhampati, S., Cutkosky, M.R., Tenebaum, J.M., and Lee, S.H., 1993, Integrating general purpose planners and specialised reasoners: case study of a hybrid planning architecture, *IEEE Trans. Syst., Man, and Cybernetics*, 23, 1503–1518.

Lee, S.H., Cutkosky, M.R., and Kambhampati, S., 1991, Incremental and interactive geometric reasoning for fixture and process planning, *Issues in Design Manufacture/Integration,* DE-Vol. 39, ASME, 7–13.

Madey, G.R., Weinroth, J., and Shah, V., 1994, Hybrid intelligent systems, tools for decision making in intelligent manufacturing, in *Artificial Neural Networks for Intelligent Manufacturing*, C.H. Dagli, Ed., 67–90, CRC Press, Boca Raton, FL.

Mäntylä, M., Opas, J., and Puhakka, J., 1989, Generative process planning of prismatic parts by feature relaxation, *Advances in Design Autom.,* B. Ranani, Ed., ASME, 49–60.

Mayer, R.J., Su, C.-J., and Keen, A.K., 1992, An integrated manufacturing planning assistant-IMPA, *J. Intelligent Manuf.*, 3, 109–122.

MetCut Research Associates Inc., 1972, *Machining Data Handbook*, Machinability Data Centre, 2nd ed.

Muthsam, H. and Mayer, C., 1990, An expert system for process planning of prismatic workpieces, *Proc. of First Conf. on Artificial Intelligence and Expert Syst. in Manuf.,* 211–220.

Nee, A.Y.C. and Senthil Kumar, A., 1991, A framework for object/rule-based automated fixture design system, *Annals of the CIRP,* 40, 147–151.

Nevrinceanu, C., 1987, Reasoning from first principles in process planning for precision machining, Ph.D. dissertation, University of Minnesota.

Nevrinceanu, C. and Donath, M., 1987, Planning with constraints in precision machining, in *Knowledge Based Expert System in Engineering: Planning and Design*, D. Sriram and R.A. Adey, Eds., 1–17, Computational Mechanics Publications.

Nevrinceanu, C., Morellas, V., and Donath, M., 1993, Automated process planning: reasoning from first principles based on geometric relation constraints, *J. Artificial Intelligence for Eng., Design, Anal. and Manuf.,* 7, 159–179.

Nordloh, H., 1994, Integration of CAD, CAM and process planning, in *Modern Manufacturing Information Control and Technology,* Zaremba, M.B. and Prasad, B., Eds., Springer-Verlag, 327–353.

Ong, S.K. and Nee, A.Y.C., 1994a, Application of fuzzy set theory to set-up planning, *Annals of the CIRP,* 43, 137–144.

Ong, S.K. and Nee, A.Y.C., 1994b, Set-up planning: a pre-planning process for CAPP and CAFP, *Proc. of the 7th Int. Conf. on Precision Eng.,* ICPE'94, 112–119, Japan.

Ong, S.K., Prombanpong, S., Zhang, Y.F., and Nee, A.Y.C. ,1993, A proposed framework for the integration of process planning with fixture planning, *Proc. of the National Symposium on Manuf. Technol.,* MANUTECH'93, 190–196, Singapore.

Opas, J., Kanerva, F., and Mäntylä, M., 1994, Automatic process plan generation in an operative process planning system, *Int. J. Prod. Res.,* 32, 1347–1363.

Ray, S.R. and Feenay, A.B., 1993, *A National Testbed for Process Planning Research*, NISTIR 5169, NIST, U.S.

Sakurai, H., 1990, Automatic set-up planning and fixture design for machining, Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, MA.

Sakurai, H., 1992, Automatic set-up planning and fixture design for machining, *J. Manuf. Syst.,* 11, 30–37.

© 2001 by CRC Press LLC

Sakurai, H. and Gossard, D. C., 1991, Geometric modelling in set-up planning and fixture design, in *Product Modelling for Computer-Aided Design and Manufacturing*, J. Turner, J. Pegna, and M. Wozny, Eds., 299–313, Elsevier Science, New York.

Salomons, O.W., van Houten, F.J.A.M., and Kals, H.J.J., 1993, Review of research in feature-based design, *J. Manuf. Syst.,* 12, 113–132.

Sarma, S.E. and Wright, P.K., 1996, Algorithms for the minimisation of setups and tool changes in "simply fixturable" components in milling, *J. Manuf. Syst.,* 15(2), 95–112.

Shah, J.J., 1988, Feature transformations between application-specific feature spaces, *Comput.-Aided Eng. J.,* 5, 247–255.

Shah, J.J., 1991, Assessment of feature technology, *Comput.-Aided Design J.,* 23, 221–233.

Shah, J.J., 1992, Features in design and manufacturing, in *Intelligent Design and Manufacturing,* A. Kusiak, Ed., 39–71, John Wiley & Sons, New York.

Shah, J.J., Mäntylä, M., and Nau, D.S., 1994, *Advances in Feature-Based Manufacturing*, Elsevier Science, New York.

Sood, S., Wright, P.K., and MacFarlane, J., 1993, Process planning: a review, *Intelligent Concurrent Design: Fundamentals, Methodology, Modelling and Practice,* ASME, DE-Vol. 66, 45–54.

Trappey, J.C. and Liu, C.R., 1990, A literature survey of fixture design automation, *Int. J. Adv. Manuf. Technol.,* 5, 240–255.

Trucks, H.E., 1987, *Designing for Economical Production,* 2nd ed., SME.

Venjara, Y., 1996, Setup savings, *Manuf. Eng.,* 117(1), 96–102.

Warnecke, H.J. and Muthsam, H., 1992, Knowledge-based systems for process planning, in *Intelligent Design and Manufacturing,* A. Kusiak, Ed., 377–396, John Wiley & Sons, New York.

Weill, R., Spur, G., and Eversheim, W., 1982, Survey of computer-aided process planning systems, *Annals of the CIRP,* 31, 539–551.

Westhoven, T.E., Chen, C.L.P., Pao, Y.-H., and LeClair, S.R.,1992, Episodal associative memory approach for sequencing interactive features in process planning, *Artificial Intelligence for Eng. Design, Analy. and Manuf. J.,* 6, 177–197.

Wiendahl, E.H.H.-P. and Fu, Z., 1992, Computer-aided analysis and planning of set-up process, *Annals of the CIRP,* 41, 497–500.

Wiendahl, H.-P. and Scholtissek, P., 1994, Management and control of complexity in manufacturing, *Annals of the CIRP,* 43, 533–540.

Willis, D., Donaldson, I.A., Ramage, A.D., and Murray, J.L., 1989, A knowledge-based system for process planning based on a solid modeller, *Comput.-Aided Eng. J.,* 21–26.

Wilson, F.W. and Holt, J.M., 1962, *Handbook of Fixture Design*, McGraw-Hill, New York.

Young, R.I.M. and Bell, R., 1991, Fixturing strategies and geometric queries in set-up planning, *Int. J. Prod. Res.,* 29, 537–550.

Yue, Y. and Murray, J.L., 1994, Validation, workpiece selection and clamping of complex 2.5D components, in *Advances in Feature Based Manufacturing*, J.J. Shah, M. Mäntylä, and D.S. Nau, Eds., 185–213, Elsevier Science, New York.

Zhang, H.-C. and Alting, L., 1994, *Computerised Manufacturing Process Planning Systems,* CRC Press, Boca Raton, FL.

Zhang, Y.F., Nee, A.Y.C., and Ong, S.K., 1995, A hybrid approach for set-up planning, *Int. J. Advanced Manuf. Technol.,* 10, 183–190.

Züst, R. and Taiber, J., 1990, Knowledge-based process planning system for prismatic workpieces in a CAD/CAM-environment, *Annals of the CIRP,* 39, 493–496.

© 2001 by CRC Press LLC

6

Object-Oriented Modeling and Simulation of Multipurpose Batch Plants

6.1 Introduction
Modeling and Simulation of Batch Plants
6.2 Multipurpose Batch Plants
6.3 Recipe-Driven Production
6.4 Requirements for Simulation Tools for Batch Processes
6.5 BASIP: The System Architecture
6.6 Graphical Model Building
6.7 Hybrid Simulation
6.8 Discrete-Event Simulation
6.9 Integrating General Simulation Tools
6.10 Handling of Resource Conflicts
6.11 Visualization
6.12 Example 1: A Simple Batch Process
6.13 Example 2: Combining Scheduling and Simulation
6.14 Discussion
6.15 Appendix: Object-Oriented Design

Sebastian Engell
Universität Dortmund

Martin Fritz
Software Design & Management AG

Konrad Wöllhaf
Universität Dortmund

6.1 Introduction

Multipurpose batch plants are used in the chemical industry to produce different products in small or medium quantities with the same equipment. This leads to economic production and flexible, fast responses to changing market requirements, especially products that exist in a large number of variants and/or are subject to frequent changes, e.g., pharmaceutics, paints, or specialty polymers.

Advanced automation techniques are needed to fully exploit the capabilities and to guarantee the safe operation of these versatile and complex plants. Recipe-driven production, now realized by many modern distributed control systems (DCS), is the key technology to achieve these goals.

© 2001 by CRC Press LLC

While for the actual operation of the plant a high degree of automation can be realized by using commercially available technologies, many high-level tasks, such as scheduling and capacity planning, still lack support by appropriate computer tools.

Modeling and Simulation of Batch Plants

Simulation has been established as a means to provide better insight into complex processes. As the complexity of batch process makes them hardly tractable for rigorous mathematical analysis, simulation is the basis for decision support systems for high-level tasks in batch plant operation and planning.

While simulation has been applied to certain aspects of batch processes in industry in the last decade, no single simulation program, not even a common approach, has been established as a standard or *de facto* standard so far. This is mainly due to the fact that no single simulation program can address the wide range of possible applications of simulation in batch production. Specialized solutions for specific areas exist, but these are not suitable for other problem domains.

A software package that tries to overcome these shortages is presented in the remainder of this contribution. Its main idea is not to develop "universal" simulator which solves all problems, but to provide a flexible framework where the components best suited for a specific application can be combined without much effort.

This framework includes a graphical model interface for the construction of simulation models in a representation the practitioner is used to, not requiring in-depth simulation knowledge.

Several different simulation strategies are provided, giving the user the freedom to choose the one which is best suited for his problem.

Visualization of the simulation results is another important aspect of a simulation package. The flexibility of the architecture makes it possible to include advanced data analysis and visualization tools.

Two examples (of Sections 6.2 and 6.3) in which simulation is used for different tasks show the applicability of the approach. The appendix gives a short introduction into the concepts and notations of object-oriented design that are used throughout this contribution.

6.2 Multipurpose Batch Plants

From a system analysis point of view, multipurpose batch plants are one of the most complex and most difficult classes of production systems. Figure 6.1 indicates the position of multipurpose batch plants in the taxonomy of production systems.

Having properties of both the process industry and the manufacturing industry, batch processes exhibit the following characteristics that do not occur in typical manufacturing processes:

- The substances involved in the production process mostly have a fluid character. No entities of produced goods can be identified trivially; transport phenomena are a crucial factor in process operation.
- Often cyclic material flow occurs, creating interdependencies that cannot be resolved easily.
- Coupled production is standard. By-products must be processed further, or at least be disposed of, in a defined way. This increases the complexity of production planning.

In contrast to classical continuous production, batch processes where the process stages are performed step by step do not have a steady state; instead, the operation conditions change frequently. In consequence, there are more degrees of freedom for control, thus again increasing the complexity.

In a single or multi-product plant, the number and the order of the process stages by which all products are produced is the same, potentially offering several parallel equivalent lines. In the (still rare) case of a multipurpose plant, in contrast there are hardly any constraints for the path a product can take through the plant, including cycles in the processing sequence. In a multiproduct plant, all batches pass through the same stages in the same order. The products are usually quite similar, e.g., paints of

© 2001 by CRC Press LLC

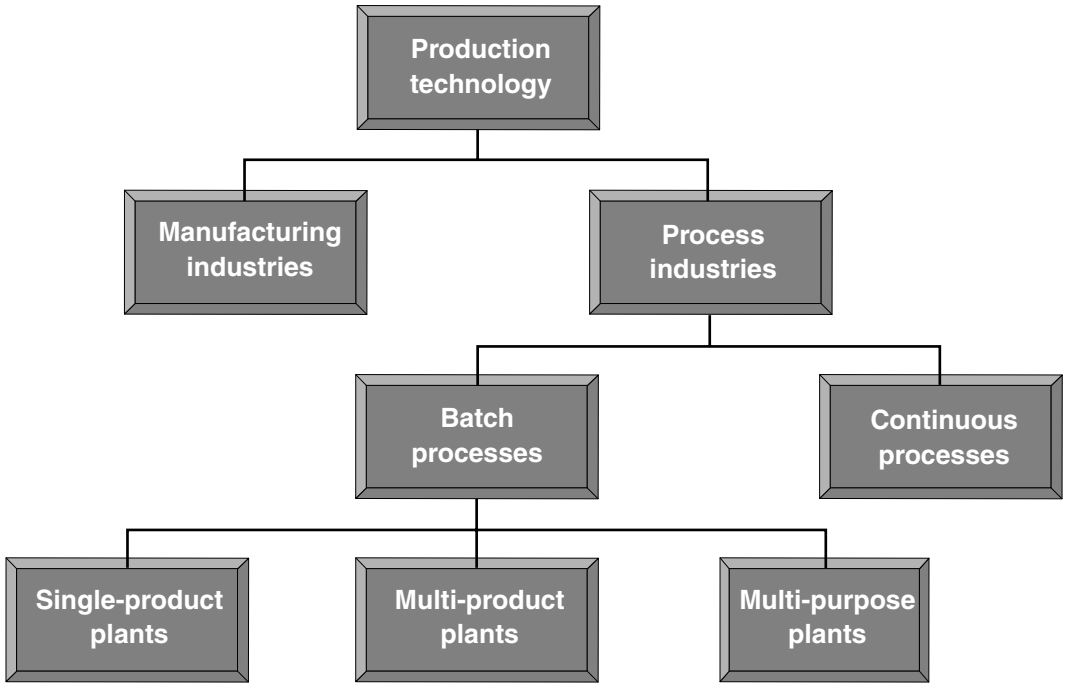


FIGURE 6.1 Taxonomy of production systems.

different colors. In a multipurpose plant, a batch can pass through the process stages of the plant in an arbitrary order.

For multiproduct and multipurpose plants, production planning has to deal with the question of what to produce when, and, in which quantities on which piece of equipment. To achieve an optimal solution, these questions have to be addressed simultaneously. In practice however, a sequential approach is mostly used.

An example of a multiproduct batch plant is given in Figure 6.2. Here, a product has to take three process stages, taking place in the equipment labeled MWx, Rx.y, and Vx, respectively. There are three identical lines where each line is split again in the middle stage. The plant equipment is strongly interconnected, so resource allocation decisions have to be taken in advance.

6.3 Recipe-Driven Production

The key idea of recipe-driven production is to describe the chemical and physical operations for the production of a specific product in a plant-independent master recipe, and to derive the actual control sequences for the execution of the production (automatically or manually) from it. This technique enables a safe and efficient usage of the plant and the production of different batches in a reproducible quality, as well as the development of well-defined instructions on how to produce a product on possibly different pieces of equipment.

By now, all major vendors of distributed control systems have extended or developed products to support recipe-driven production. Nevertheless, especially in older plants, a purely manual operation of the plant is still a common case, and often a fraction of the operations, especially solids processing, must be performed manually.

Quite early, steps to standardize recipe-driven production were taken. A leading role was played by NAMUR, a German industrial consortium for process automation, which in 1992 released its recommendation NE 33, "Requirements for Batch Control Systems." [NAMUR-Arbeitskreis, 1992] Partially based on this work, the ISA Technical Committee SP88, in cooperation with IEC and ISO groups, has been working on

© 2001 by CRC Press LLC

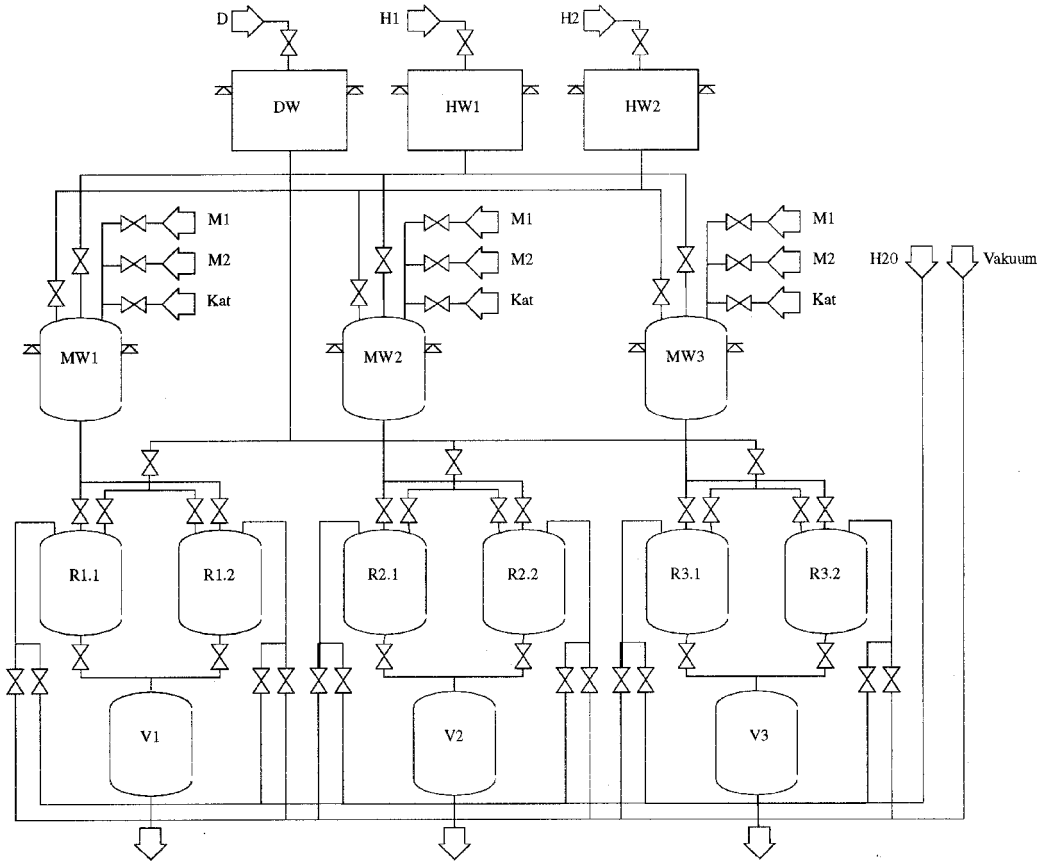


FIGURE 6.2 Example of a multiproduct batch plant.

an international standard for batch process control, whose first part, S88.01, "Models and Terminology" was released in 1995 [ISA, 1994]. The second part, "Data Structures and Guidelines" has not been published yet.

The intent of these standards is to define reference models and notions to improve the planning and operation of batch plants, and to facilitate communication between all participants with their different backgrounds from chemistry, process development, and control engineering.

Central to S88 is the so-called control activity or "cactus" model (see Figure 6.3). It structures the activities involved in batch control into plant management functions (upper level) and batch processing (lower levels). The S88 model differentiates between the equipment on which the process is performed, the process description captured in recipes, and the process itself.

The equipment needed for batch production is modeled by functional objects and is structured hierarchically as follows [Haxthausen, 1995].

- On the lowest level is the control module. It consists of a device together with the controls needed to perform its functions. A typical example is a valve. Its minimal control equipment consists of a handle to open and close it manually.
- Control modules can be composed into equipment modules which are defined lay being able to perform a processing task. They need a coordinating logic, called phases, to coordinate the process actions. In terms of NE 33, these are called technical functions.
- The level of a unit corresponds to a batch. There is never more than one batch in a unit, and a unit usually processes the whole batch (though a batch may be split into several units). A unit can

© 2001 by CRC Press LLC

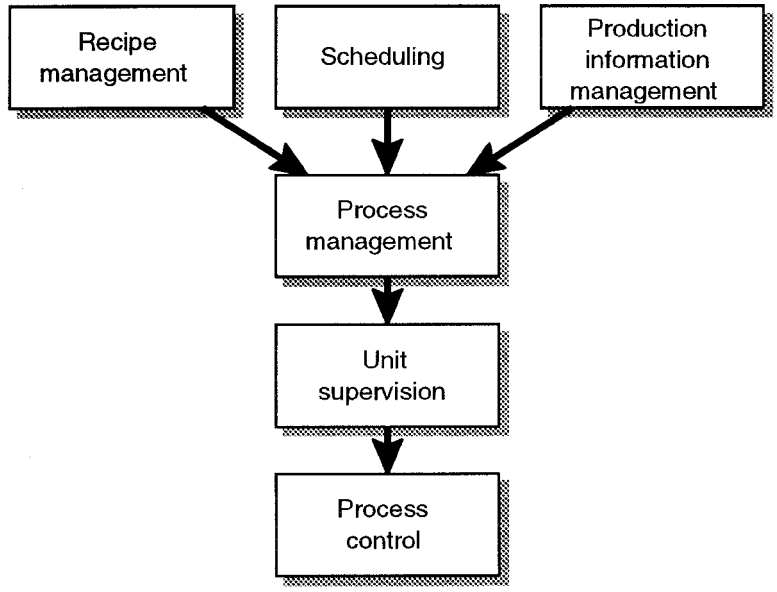


FIGURE 6.3 S88.01 control activity model after [ISA, 1994].

execute phases, operations, and unit procedures (which in turn form a hierarchy), and is responsible for the coordination of these steps.

- The process cell, finally, contains all the equipment needed to produce a batch. It can define a number of procedures.

To produce more than one product on the same plant, recipes have to be introduced to specify how each product is produced. The recipe tells the process cell which of its procedures must be executed in which order. This mapping may be performed on different levels, thus, the recipe can refer to procedures, unit procedures, operations, or phases. This requires that the recipe has a hierarchical structure itself (see Figure 6.4).

Recipes are also categorized by their degree of abstraction. A control recipe describes the execution of an individual batch. It is derived from a master recipe, which may be used on different equipment of the process cell, or even on different process cells, as long as they fulfill the process requirements. Master recipes may be derived from more abstract site or general *recipes* which are defined in pure processing terms with no relation to specific equipment.

6.4 Requirements for Simulation Tools for Batch Processes

There are many possible applications of simulation to batch processes, as mentioned in the introduction. The following listing is not comprehensive, but shows the wide range of potential applications:

- Operator training. A training simulator replaces the real plant in a real process control environment.
- Case studies. The investigation of hazardous or otherwise unusual or unwanted situations is only possible by simulation.
- Plant planning. Exploring the necessary capacities of future or the bottlenecks of existing plants.
- Recipe design. Testing and validating of newly developed or modified recipes before applying them to the plant.

© 2001 by CRC Press LLC

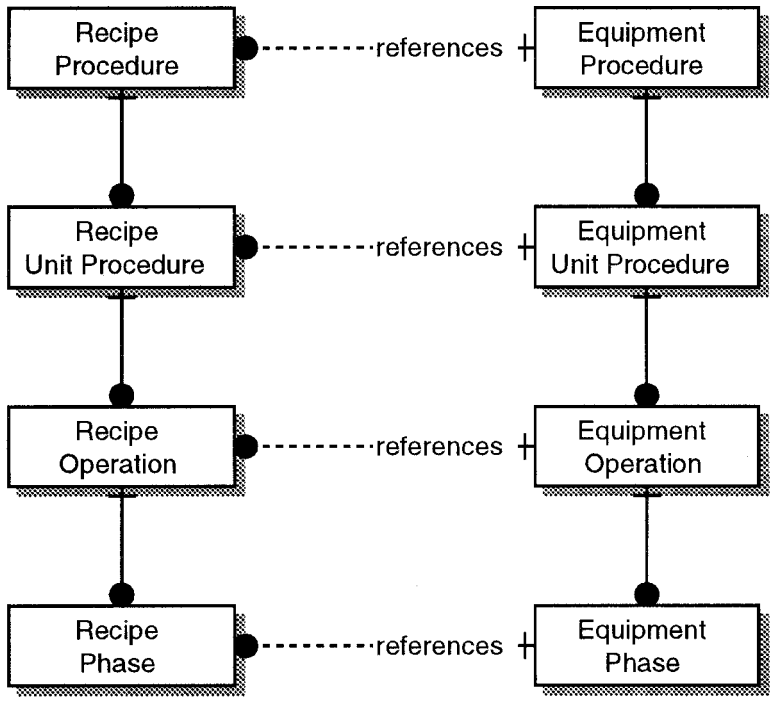


FIGURE 6.4 Possible mappings of recipe to equipment after [ISA, 1994].

- Scheduling and production planning. Investigating possible resource allocation conflicts or deadlocks when executing several recipes sequentially or in parallel.
- Plant-wide logistics. Examining the material flow through the whole production, from purchasing to distribution.

Hence, there are different requirements on the simulation and the model used:

- The necessary level of detail of the model depends on the accuracy needed for the given application. While a training simulator requires a rigorous evaluation of the chemical and physical properties of the process, for logistics it may even be sufficient to model a reactor as a device that is just either idle or busy.
- The performance of a simulation run may be crucial for the application as well. While real-time simulation, as needed for training simulators, is usually not too hard to achieve because the process dynamics usually are rather slow, most applications demand that the simulation is running much faster than real time. For applications in production planning, response times of seconds, or a few minutes, can be demanded. This may not be achievable with detailed models and continuous simulation approaches.

Thus, performance and accuracy are normally opposing goals, which means that in practice, the compromise which is the best fit for the specific problem has to be chosen.

There are more requirements that apply to all applications:

- User support. A simulation tool should be easy to use and not require expert knowledge in simulation techniques or operating system peculiarities. Therefore, a tool should have a state-of-the-art graphical user interface enabling graphical model building, configuration, and execution of simulation runs, and monitoring and analysis of simulation results within a single environment.

© 2001 by CRC Press LLC

To achieve widespread use, it must be available on popular hardware/operating system platforms in the PC and workstation market.

- Conformance with existing standards. Model building should be possible using the same concepts and techniques as for real applications. That is, the plant should be specified in a flow sheet style, while the creation of a production schedule is done in terms defined in ISA S88, i.e., master and control recipes, operations, phases, etc. No translation of notions of the real world into those of the simulation system should be necessary.
- Integration into existing information systems structures. A successfully used simulation system cannot be an isolated solution, but has to be incorporated into the enterprise's information systems. This may imply that the simulation makes use of data collected by a process monitoring system, or exports its results directly into a management support system for comparison with actual plant data.

No single tool can fulfill all of these requirements. What is needed instead is a software architecture that can be configured and customized to provide the best solution to the given problem in the given environment. The most important property of such a framework is flexibility; it must be able to adapt itself to possible applications under certain constraints that are not known beforehand.

6.5 BASIP: The System Architecture

A prototype of such a flexible, integrated modeling and simulation environment has been developed at the Process Control Group at the University of Dortmund Germany [Wöllhaf, 1995] and [Fritz and Engell, 1997]. The resulting software system is called BASIP (Batch Simulation Package). Its structure is shown in Figure 6.5. To achieve flexibility, the software package is divided into three largely independent components for model building, simulation, and visualization. They can be managed and configured via a single user interface.

The architecture reflects the object-oriented principles which were used for its design (as well as for its implementation). Interconnections between the single components are minimized and realized by abstract interfaces. Thus, the components are encapsulated and can be exchanged by alternative implementations

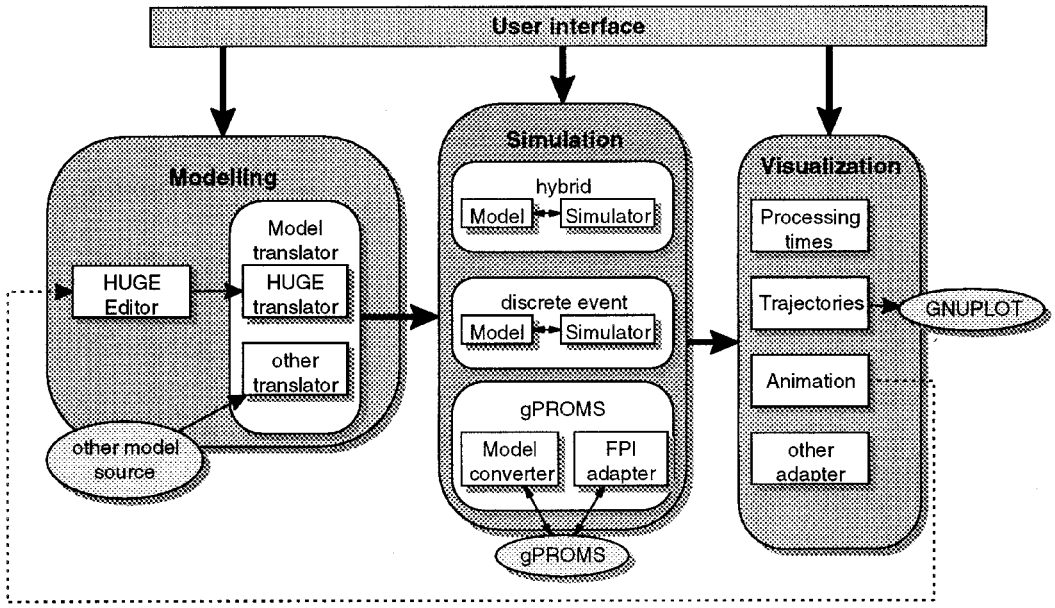


FIGURE 6.5 The BASIP system architecture.

© 2001 by CRC Press LLC

that only have to fulfill the minimum requirements which the other components put on them. This facilitates the integration into other software systems and the adaptation of the system to a particular application [Fritz and Engell, 1997].

On the modeling side, the package contains several graphical editors to easily create simulation models (Cf. Section 6). The simulation engine, however, which uses the models generated with these editors, does not rely on the particular database structure or file format used to store these models. Instead, it accesses the model via a model translation component that transforms the model files into high-level abstractions. If model data from other sources must be used, e.g., to directly simulate recipes stored in the recipe management system of a DCS, only an alternative implementation of the model translator has to be generated; the simulator remains unchanged.

Analogously, the system does not put any restrictions on the way the simulation results are displayed to the user. There is only a specification of what sort of data is available in what format. A visualization component (or outputClient in BASIP's terminology) then can be added that processes this data in its own manner. Most of the outputClients predefined in the system do not perform a lot of processing themselves, but rather act as bridges to external data visualization or analysis software.

Even the backbone of the system, the simulator, is exchangeable. Thus, different simulation methodologies or different numerical solution strategies can be applied, either to increase the accuracy of the simulation results, or to choose the simulator with the best performance for the given application.

6.6 Graphical Model Building

A BASIP model is structured as shown in Figure 6.6. What is actually simulated is a collection of control recipes. A control recipe contains a reference to the corresponding master recipe and to the plant on which production is executed or simulated. Furthermore, it contains the mapping of the phases of the master recipe to equipment phases of the plant. Thus, the mapping is realized on the lowest level offered by the S88 model.

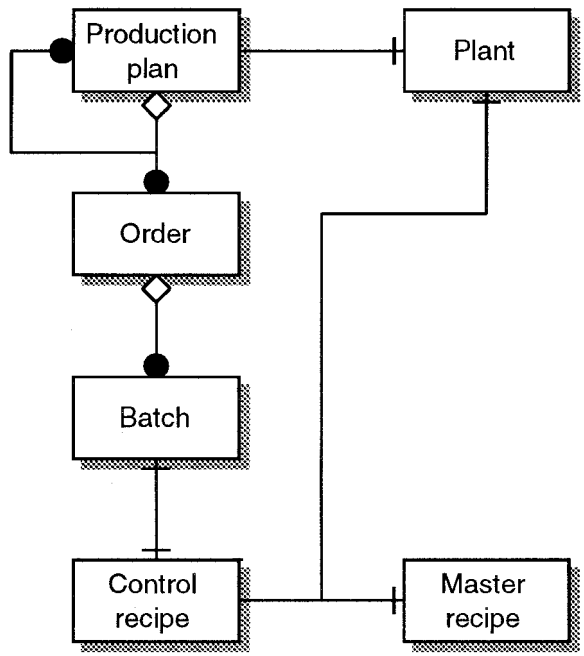


FIGURE 6.6 The BASIP model structure.

© 2001 by CRC Press LLC

Control recipes do not contain information about the time at which they are executed. As a result, they can be used to produce more than one batch. This is a slight deviation from the S88 notion, in which a control recipe is linked to exactly one batch. Here, information about starting times is transferred to a separate batch object which also contains information about the products produced by this batch and the feed materials needed. Batches can be put together into orders to reflect related production runs, e.g., of the same product or for the saline customer. On the highest level, a production plan contains all orders for a given period of time. It can be composed of other production plans to structure them with respect to time or space.

For all model components, graphical model editors are provided. Of particular interest are the editors for master recipes and plants, as they offer true graphical modeling instead of text-oriented dialog windows. While plants are represented in a flow sheet style, recipes are modeled as sequential function charts [IEC, 1988] as recommended by both NAMUR and ISA.

Both representation forms can be abstracted to (directed) graphs with different types of nodes and edges. A graphical editor to manipulate graph structures must have some advanced features compared to standard drawing programs because they must preserve the semantic relations between nodes and edges. The editors for plant and master recipes are based on a common foundation: a framework for graph editors called HUGE (hierarchical universal graph editor) [Fritz, Schultz, and Wöllhaf, 1995]. It provides the following features for any application derived from it.

- Creating, deleting, connecting, and moving of nodes and edges.
- Grouping of nodes and edges to superblocks (subgraphs), thus enabling the creation of hierarchical structures.
- Unrestricted undo and redo of user operations.
- A clipboard for cut, copy, and paste.
- Persistence mechanisms to store and retrieve models to/from disk.

A sequential function chart that is used for modeling master recipes consists of a sequence of steps and transitions. Every step has a corresponding action that refers to certain type of predefined recipe phases that can be parameterized. For syntactical reasons, the null action (no operation) is also admitted.

The following two types of branches are used for the structuring of recipes.

- Alternative branches ("or"-branches) force the flow of control to follow exactly one of the branches, depending on the transitions at the beginning of the branch (see below).
- Parallel branches ("and"-branches) define a parallel execution of all branches between two synchronization marks (parallel lines).

Each step is followed by a transition containing a condition that determines when the transition switches. If the condition is true, the preceding steps are deactivated and the subsequent ones are activated. In analogy to the switching rules of Petri nets, a transition can only switch if all of its preceding steps are active. The conditions in transitions can refer to time or to physical quantities of the plant, e.g., the level or the temperature of a vessel.

Figure 6.7 shows the realization of this structure in BaSiP's master recipe editor RezEdit. The restrictive syntax of a sequential function chart allows for an automatic syntax checking, thus prohibiting any syntactical errors. Also, the layout of the recipe is generated automatically, saving much of the user's time otherwise needed to produce "nice" drawings. The recipe editor furthermore comprises a dialog box to specify the products, educts, and their default quantities.

For the plant model, care must be taken regarding on what level of detail modeling should take place. Continuous simulation techniques require the description of a model block using a set of differential equations. This notion, however, does not make sense for a discrete event simulator. Furthermore, creating models in terms of differential equations (or other modeling paradigms like finite state machines) requires expert knowledge that should not be necessary to use the simulator.

Therefore, BaSiP models are split into a generic and a specific part. Only the generic part, i.e., the information that is necessary for all simulation approaches, is contained in the models created with the

© 2001 by CRC Press LLC

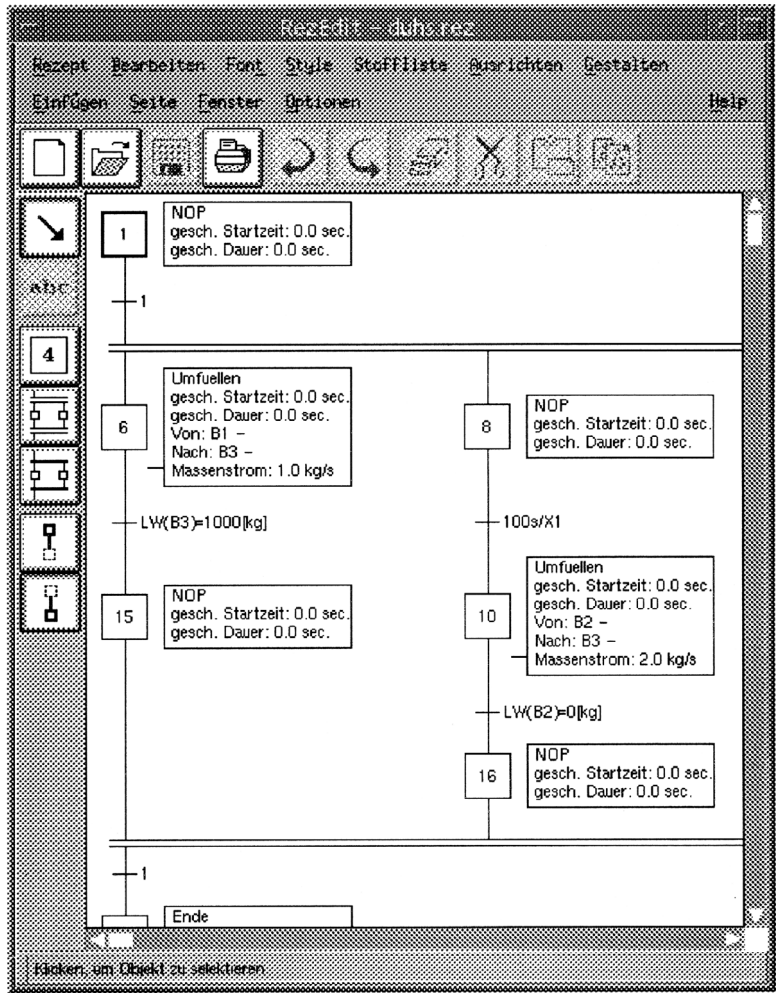


FIGURE 6.7 The master recipe editor REZEDIT.

model editors. Generic information contains the model structure, i.e., which equipment is connected with each other as well as equipment parameters, e.g., vessel sizes or minimum and maximum flow rates of a pump. The specific part, different from simulator to simulator, is contained in a model library that is specific for each simulator.

With the plant editor PLANTEDIT, shown in Figure 6.8, a flow sheet model of the plant can be built. Equipment can be chosen from a library of equipment types and is connected by material or energy flows. Furthermore, equipment phases (technical functions) as, e.g., dosing, heating, stirring, etc. can be assigned to the equipment items. Each equipment item and each phase has a number of parameters that can be set by the user.

Since an automatic routing of connections would not lead to satisfactory results in all cases, the plant layout is under the user's control, but is supported by several layout help functions.

6.7 Hybrid Simulation

For some applications in batch process simulation, the dynamics of the plant equipment and the substances contained cannot be ignored. In contrast to the manufacturing industries, purely discrete models cannot provide the required accuracy. Thus, the dynamics are described using differential equation systems, the

© 2001 by CRC Press LLC

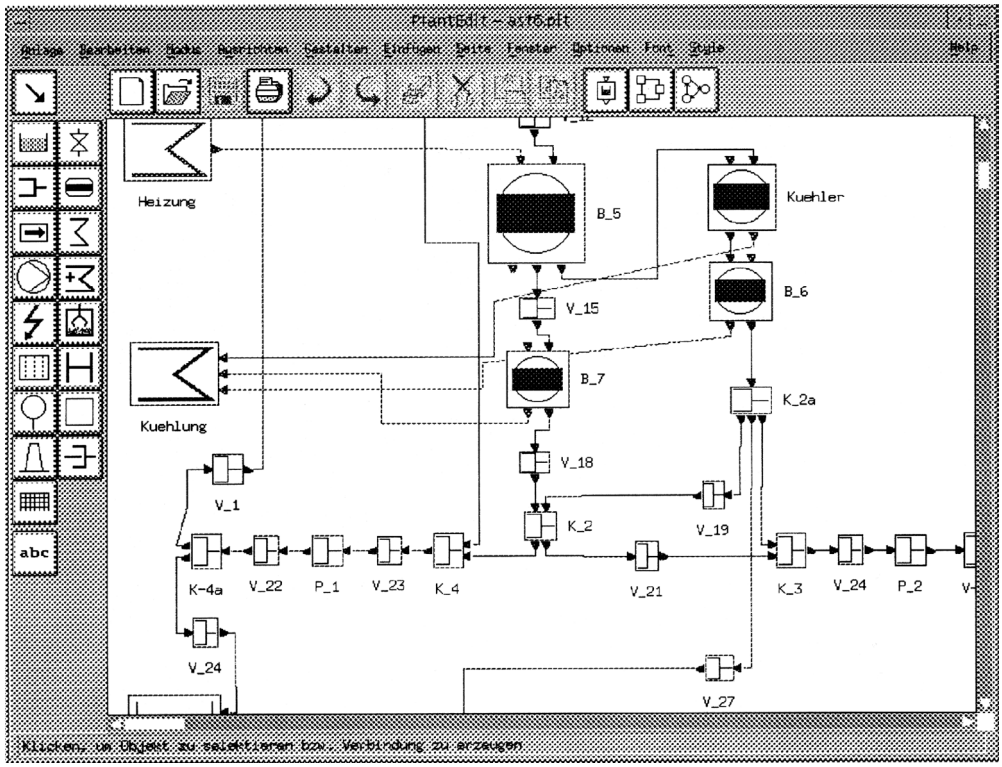


FIGURE 6.8 The plant editor PLANTEDIT.

standard modeling paradigm for continuous systems. So far, only systems of ordinary differential equations (ODEs) are handled by the numerical solver. The more general case of differential-algebraic equations (DAEs) is suited better to model certain phenomena that occur in batch process dynamics, but solvers for these systems require more computational effort, especially in the case of hybrid systems.

On the other hand, batch processes are discontinuous by nature, i.e., discontinuous changes in the system inputs and of the system's structure frequently occur. From a system-theoretic point of view, batch processes belong to the class of hybrid systems. Such systems have both continuous and discrete dynamics which are closely coupled [Engell, 1997].

Neither conventional continuous nor discrete event simulation techniques can handle such systems. New simulation approaches are necessary for an adequate treatment of hybrid systems.

One approach taken in BASIP is to partition the simulation time into intervals in which the system structure remains constant and the system inputs are continuous. The resulting differential equation system can then be solved by a standard numerical solver. The key problem of this approach is, of course, to find the right partition. This problem is also known as event detection. An event is a point of time where a discontinuity occurs.

Events can be divided into two categories: if the time when an event occurs is known in advance, it is called a time event. A time event is induced, e.g., by a recipe step with a fixed duration. Time events are easy to handle for a simulation program. For state events, on the other hand, the point of time depends on the values of state variables. Special methods are necessary to determine the exact time at which the event occurs. This problem corresponds to finding the roots of a switching function that is, in general, given only implicitly. Neither time nor state events can be determined before the simulation starts. Thus, event detection is an integral part of a simulation algorithm for hybrid systems [Engell et al., 1995].

The basic structure of the simulation algorithm is shown in Figure 6.9. In the first step after initialization, the state vector $(\underline{x}, \dot{\underline{x}})$, consisting of the state variables and their derivatives, is reconfigured, i.e., the

© 2001 by CRC Press LLC

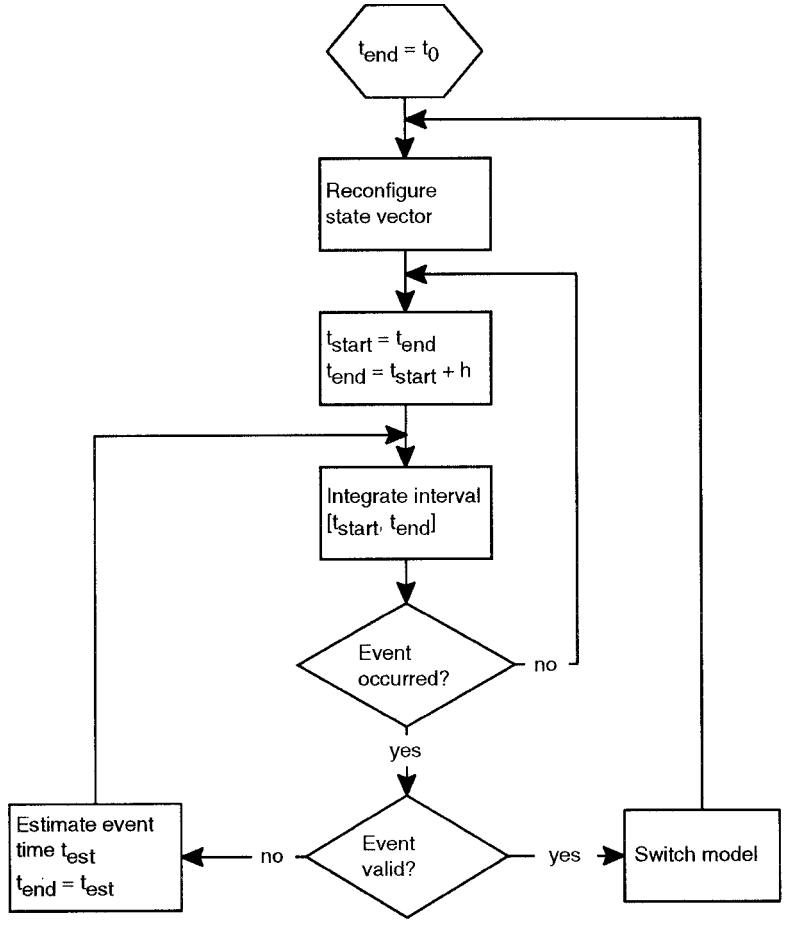


FIGURE 6.9 The algorithm for hybrid simulation.

currently active state variables are determined and inserted in the vector. This step is repeated after each successful event detection step. Note that in this reconfiguration step the number of state variables in the state vector can change and discontinuities in the values of state variables are allowed. By this dynamic reconfiguration, the size of the state vector is kept minimal, eliminating unnecessary, i.e., not affected state variables; thus keeping numerical integration efficient [Wöllhaf et al., 1996]. In most simulation approaches, the state vector is set up once at initialization time and contains all state variables, whether or not they are active in a specific interval.

Next, the integration interval is set using a maximum step width h. The integration routine in the next step evaluates the state vector at the end of this interval. It may use internally, a smaller, or even variable step width.

After every integration step, it must be checked to determine whether an event has occurred. How this is done is explained below. If an integration method uses several internal steps to integrate the interval, it might be useful to repeat the event, checking after every internal step. If no event has occurred, the next interval can be integrated; otherwise, the validity of the event is checked, i.e., it is tested, whether the time of the event coincides with the end of the interval. Although this is not equivalent in all cases, it usually suffices to test if the value of the threshold crossed is still sufficiently close to zero. If this is the case, the event is valid and the model can be switched, i.e., the discrete part of the model is re-evaluated. Otherwise, one tries to get an estimate for the time of the first event (there might have occurred more than one event in an interval) that occurred. This can be done by an interpolation using, e.g., *regula falsi*, or, if this fails

© 2001 by CRC Press LLC

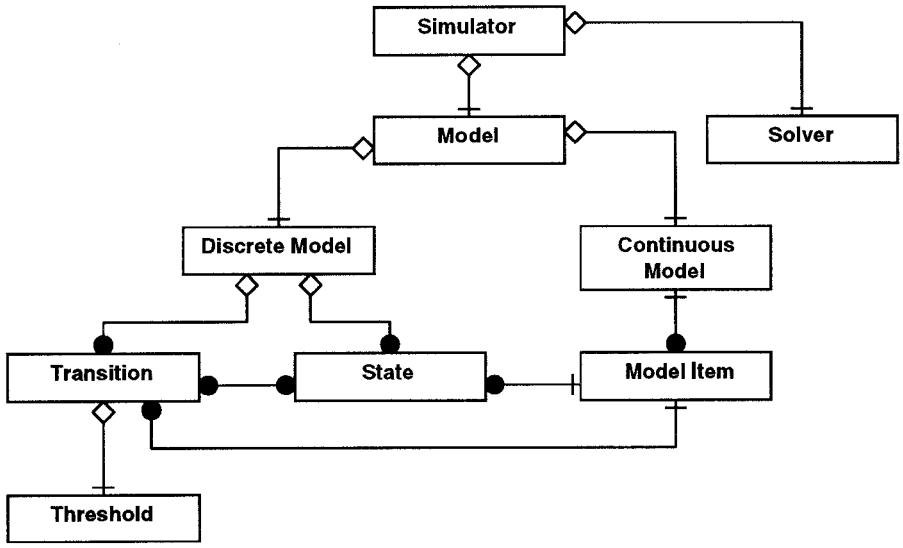


FIGURE 6.10 The object model of the hybrid simulator.

due to strong nonlinearities, the interval is simply halved. The end time is set to the estimated time and the interval is integrated again. If the estimation was good enough, the event time will coincide with the end of the interval and the event will be declared valid. Otherwise, this procedure has to be iterated until the interval end is close enough to the event time, or the interval width is below a predefined limit.

Using the object-oriented approach, this algorithm call be implemented using a set of cooperating objects. The object model of the hybrid simulator is depicted in Figure 6.10. The driving force in this structure is the simulator. It performs its task using two components: a numerical solver, which carries out the integration using a numerical integration method which is entirely encapsulated, and, on the model. The model is responsible for configuring the state vector, evaluating the equations, event checking, and model switching.

For simulation purposes, it does not make sense to preserve the division of the model into plant and recipes; instead, the model has a continuous and a discrete part. These two classification schemes are not equivalent, since the plant model itself may have a continuous, as well as, a discrete part, the latter stemming from physical discontinuities like phase transitions front liquid to gaseous, a vessel which runs empty, or a switched controller.

In the continuous model part, there is a one-to-one mapping of equipment and phases defined in the plant model to simulation model items. These objects are augmented in the simulation by differential equations describing their behavior. Each model item manages its own state variables and implements a call method that returns the derivatives of its state variables depending on the current system state.

Model items are structured further to reflect the plant structure, thus facilitating modifications or extensions. Figure 6.11 shows the object model. Model items are divided into equipment and equipment phases. Equipment phases are the active elements in the simulation; they force else calculation of the derivatives of the equipment they are operating on, depending on the current plant state. Equipment objects are passive elements; their state variables are only recalculated if at least one phase operates on them. There may be exceptions, e.g., when a vessel cools down due to thermal losses to the environment. These situations have to be handled separately.

The most important type of equipment is a vessel. It contains a mixture of (possibly) different substances. The state variables for a vessel are its enthalpy and the masses of each substance it contains, separated by state, i.e., liquid and gaseous. In contrast, a resource serves as a system boundary, providing an unlimited reservoir of a certain substance.

© 2001 by CRC Press LLC

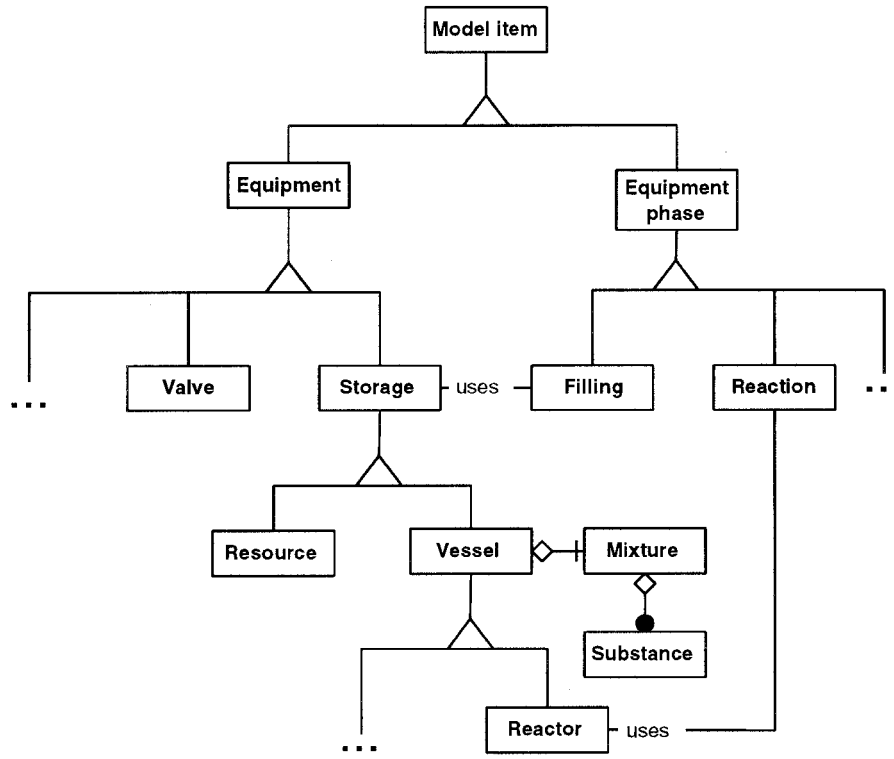


FIGURE 6.11 The object model of the model items.

The discrete part of the model consists of a state graph containing states and transitions. The state graph consists of several parallel independent subgraphs with local states. Such a subsystem can be a recipe, where as the state is composed of the steps which are currently active (and the transitions correspond naturally to the transitions of the recipe). It can also describe discrete behaviors of the plant equipment and the substances contained, e.g., the filling level of a vessel using the states empty, medium, and full, or the aggregate state of a substance. In both cases, the transitions between the states are expressed by conditions for the values of physical quantities.

Transitions make use of threshold objects which describe the switching conditions like $A(t) > B(t)$ in the form of switching functions $h(t) = A(t) - B(t)$. In the event detection step of the simulation algorithm, the state graph has to be evaluated. For that purpose, the transitions following all active states are checked to determine whether the switching function has changed its sign.

In the model switching step, each switching of a transition deactivates its predecessors and activates its successors. Each affected state notifies its corresponding model item of the change so that subsequent model evaluations can reflect the new state.

6.8 Discrete-Event Simulation

Another approach to cope with the hybrid nature of batch processes relies on the discrete-event modeling and simulation paradigm. In a discrete-event model, a system is always in one of a (finite or infinite) number of states. A transition to another state is possible if a condition depending on external inputs or on the internal dynamics of the system is fulfilled. A state change is called an event. Common representations for discrete-event systems are finite automata, state charts, Petri nets, or queuing systems.

© 2001 by CRC Press LLC

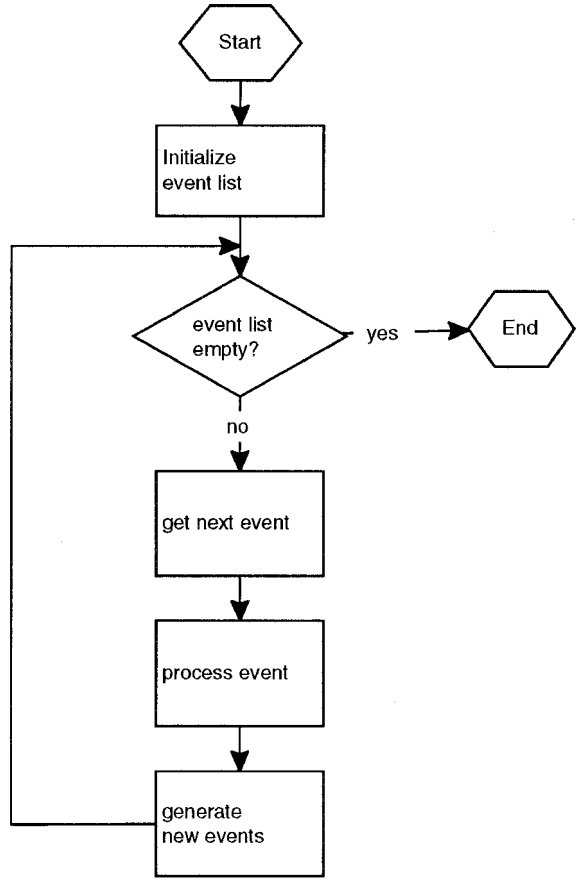


FIGURE 6.12 Algorithm for discrete-event simulation.

The purpose of a discrete-event simulator is to determine the order of state changes which a system performs due to its internal dynamics and external inputs, starting from a given initial state. In timed models, the duration for which the system remains in each state is also of interest. Thus, a discrete-event simulator is only interested in the system at the points of time where state changes occur. Figure 6.12 shows the simulation algorithm of a conventional discrete-event simulator. Central to the algorithm is the event list, a data structure containing all future events sorted by the time they are expected to occur. The simulation algorithm mainly consists of a loop that takes the first event in the event list, removes it from the list, sets the simulation time to the time of the event, and performs the action associated with this event. This action may involve the generation of new events that are inserted in the event list. Their associated time must not lie before the current simulation time. If the event list is empty, or some other termination condition is fulfilled, e.g., a maximum simulation time is reached, the simulation is terminated.

It is noteworthy that the notion of event generation that usually takes the main part of a discrete-event simulator, is semantically equivalent to event detection, introduced in the section about hybrid simulation. The difference lies rather in the point of view, whether events are something inherently contained in the system model and have to be discovered, or whether they are created by the simulation program itself. Thus, both approaches have more in common than the traditional dichotomy between continuous and discrete-event simulation implies. The term event-based simulation could be applied as a generic term to both approaches.

© 2001 by CRC Press LLC

In the case of batch processes, the simulation is mainly driven by the execution of the recipes. Recipes are nearly identical in their syntax and semantics to a class of Petri nets, so called predicate/transition (Pr/T) nets [Genrich, Hanisch, and Wöllhaf, 1994]. Steps in the recipes correspond to states of the system. The events the simulator has to generate are mainly the switching times of the recipe's transitions; there may be additional internal events to reflect structure switches in the plant.

Thus, if a discrete-event simulator is able to compute the points of time when transition in the recipe occur exactly, combined with an adequate checking of threshold crossings in the plant, no loss of accuracy, compared to the hybrid/continuous approach presented in the previous section, occurs. The only difference is a reduced output generation because no information is available about the state of the system between events. This is usually offset by a considerable acceleration of the simulation with respect to CPU time, not seldom shortening answering times by orders of magnitude.

The calculation of event times, however, is not a trivial task, and is, in fact, impossible for the general case. This is due to the fact that a condition in a transition can refer to any quantity in the plant, e.g., temperature, level, or pressure of a vessel. The semantics of a recipe do not even demand that this quantity has to be related to the preceding steps, e.g., a condition following a step that executes the phase "filling from vessel A to vessel B" could refer to the level of vessel C. Though cases like this often indicate an error in recipe design, a unique association of phase types to transition conditions is not always possible. As a consequence, the whole plant state had to be evaluated for a point of time not known in advance. Nevertheless, for many cases occurring in practice, at least an approximate solution can be computed efficiently.

Figure 6.13 illustrates the structure of the components of the discrete-event simulator. As in the case of hybrid simulation, the main component is the simulator which works on a model. The structure of the model corresponds closely to the real world: it consists of a plant, containing equipment and EquipmentPhases. Not shown in the diagram is the further subdivision of equipment into vessels, valves, etc. Analogously, several specific phases like filling or reaction are derived from EquipmentPhase.

The other part of the model consists of a set of recipes consisting of steps and transitions linked according to the recipe structure. Each step has an associated RecipePhase from which specialized types are derived in analogy to the EquipmentPhases class hierarchy. Via the control recipe, RecipePhases are mapped to EquipmentPhases.

Two types of events link the EventList to the model. StepEvents are associated with a step in the recipe. They are generated for all steps following a switching transition and for the initial step of each recipe. When activated, i.e., being the first in the event list, they try to compute their duration. For that purpose, they mark the associated equipment phase as active and ask the equipment referenced in the condition of the subsequent transition when this condition will be fulfilled. The equipment knows the equipment

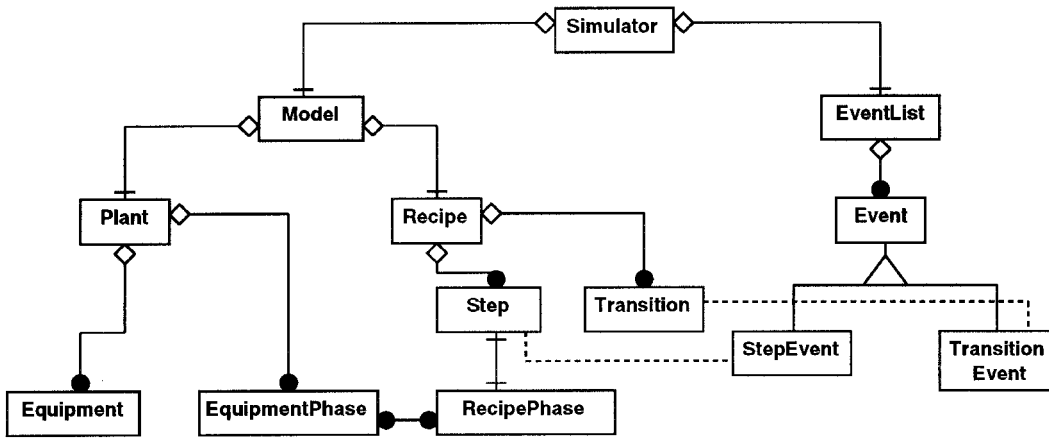


FIGURE 6.13 The object model of the discrete-event simulator.

© 2001 by CRC Press LLC

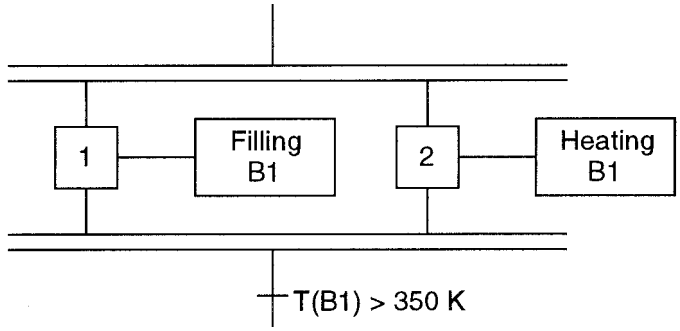


FIGURE 6.14 Example for the calculation of event times.

phases currently operating on it and tries to compute the time the condition is fulfilled. If it succeeds, it returns the time, and a corresponding TransitionEvent is generated for that time. If the computation fails, however, it might be possible to compute the event time later. In the example in Figure 6.14, Steps 1 and 2 of the recipe are executed in parallel. In a sequential simulator, however, the activation of the associated recipe phases is executed sequentially (although both events happen at the same simulation time). If Step 1 is executed first, the event time of the subsequent transition cannot be determined, since it refers to the temperature of Vessel B1, which is not affected by the filling operation. Thus, a TransitionEvent for the transition following is inserted at the last possible time, e.g., the end of the simulation. When Step 2 is activated, however, the time of the TransitionEvent can be calculated, and its position in the event list is updated accordingly.

As a consequence, the start or end of any operation in the plant potentially affects the time of all TransitionEvents pending in the EventList. Thus, after each activation of an event, these times have to be recalculated. If a TransitionEvent is activated, it is first checked to determine whether the transition condition is really fulfilled. Although that should actually always be the case, in some cases dealing with branches or transitions containing logical expressions, only an earliest possible event time can be computed. This is still inserted in the EventList because an event could be missed. If the condition is not fulfilled, the TransitionEvent is discarded, assuming that the EventList will contain another TransitionEvent for the same transition at a later time.

To compute the time when an event occurs, an equipment item has to consider its own state and the phases currently operating on it. Similar to the hybrid simulation, for a vessel the state is modeled by the enthalpy and the masses of the substances contained. To accomplish the prediction of switching times, it is assumed that all operations are linear, i.e., are changing the state variables by constant rates. This assumption is legitimate when the process is viewed from a sufficiently high level. Filling operations using constantly pumping pumps or electrical heating satisfy these assumptions. Two important exceptions exist that require special treatment

1. Simultaneous filling operations. If a vessel is filled and emptied simultaneously, the concentrations of the substances contained cannot be described using linear functions. The resulting differential equation, however, has an analytical solution. Using this, an average concentration can be computed and the assumption of constant rates can be maintained.
2. Phase transitions. The transition of a substance's state from liquid to gaseous or vice versa cannot be described with linear functions either. However, the resulting phenomena can be partitioned into intervals for which the assumptions of constant rates still hold, e.g., in a simple model, the vaporization of a substance within a mixture can be divided into three different phases (a) in the first phase, the mixture is heated to the boiling point of the lowest boiling substance. The mass remains constant, the enthalpy is increased by the heat energy supplied, (b) in the second phase,

© 2001 by CRC Press LLC

the substance evaporates, decreasing the mass of the mixture by a constant rate (gaseous phases are assumed to leave the system). The temperature remains constant, but the enthalpy of the mixture is changing with a different rate as before, and (c) after the boiling component has completely vanished, the mass again remains constant and the enthalpy is changing with the same rate as in the first phase, increasing the temperature again.

As these different phases are not triggered by external events, the equipment itself has to take this switching behavior into account. When computing the duration of an operation, it therefore generates, if necessary, so-called internal events that mark the beginning or the end or the evaporation phase. In the evaporating phase, an internal operation that adjusts the material and energy balances accordingly is activated.

6.9 Integrating General Simulation Tools

The strength of the BASIP architecture is its flexibility which enables the incorporation of other components. An example is the integration of the simulation tool gPROMS.

gPROMS (general process modelling system) is a multipurpose modeling, simulation, and optimization tool for hybrid systems [Barton and Pantelides, 1994]. It supports modular modeling and provides numerical solution routines for systems of differential-algebraic equations (DAE), as well as for partial differential equations. gPROMS provides its own data visualization tool, gRMS (gPROMS result management system). The optimization component of gPROMS, gOPT, is not used in BASIP so far.

A disadvantage of gPROMS is its purely textual model interface. gPROMS models are formulated in the gPROMS modeling language and stored in files in ASCII format which serve as input for the simulator. Despite the modular structure of this language, an in-depth knowledge of the language syntax and semantics, as well as the the underlying modeling and simulation principles, are necessary to create gPROMS models.

In consequence, batch processes cannot be modeled in terms of recipes or equipment; instead, these have to be translated into the gPROMS language elements. Thus, it is an obvious idea to combine the graphical modeling facilities of BASIP with the numerical power of gPROMS. The resulting structure is shown in Figure 6.15.

The idea is to automatically generate gPROMS input files from models created with the BASIP graphical editors. The component responsible for this task is the model converter who reads BASIP model descriptions and produces a file containing the model in the gPROMS modeling language. Because BASIP models do not contain simulator-specific information, the converter has to add additional information taken from a library of basic blocks, i.e., equipment and phase types.

The converter is internally divided into two main parts: a converter for the plant, and one for the recipes. The plant converter maps equipment types to gPROMS models. A gPROMS model contains variables parameters and equations connecting these. A special kind of variables are STREAMs that connect variables of the model to those of other models. Equations in gPROMS can depend on the value of other variables, thus switched systems can be described. Models can be composed of other models,

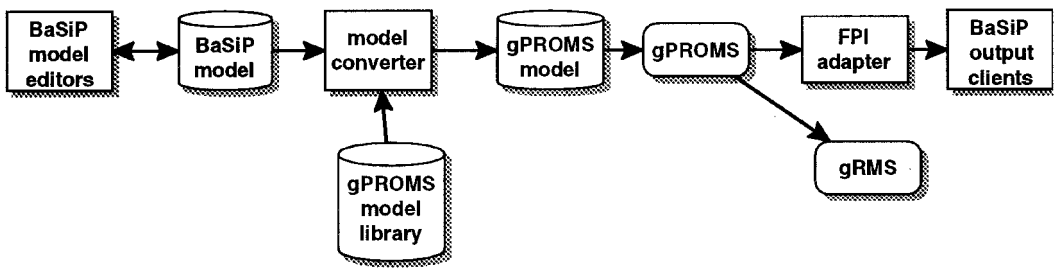


FIGURE 6.15 Integration of gPROMS into BASIP.

© 2001 by CRC Press LLC

yielding a hierarchical structure with the model of the whole plant on top. The task of the plant converter is to pick the right models from the model library, adjusting their parameters, and setting up the connections between the equipment by connecting the streams of the corresponding models.

The recipe converter maps the recipes using the schedule section of gPROMS. The schedule describes the operations performed on the models. A schedule can be composed of sequential and parallel paths. The basic element of a schedule is a task which is either primitive, e.g., the setting of a discrete variable of a model, or complex, i.e., composed of other tasks. The recipe converter has to translate the recipe structure into these language elements. To translate the transition conditions, it makes use of programming language constructs like if…then or continue until.

The results can be displayed using gPROMS native gRMS interface or via an adapter that uses gPROMS foreign process interface (FPI). This adapter takes the intermediate results produced by gPROMS and forwards them to the output clients in the specified format (see Section 6.11).

6.10 Handling of Resource Conflicts

Another aspect which is important especially when simulating several recipes in parallel, is the resolution of resource conflicts. A resource conflict occurs when two or more equipment phases need access to the same equipment to perform their operation. Several cases have to be distinguished:

- The resource (equipment) has exclusive usage. Only one operation at a time can use it.
- The resource is sharable. Several operations may use it simultaneously. There can be different types of restrictions on the simultaneous use of sharable resources (a) the resource may have a limited capacity, e.g., a maximum power available for the heating of different reactors, and (b) the resource can only be used in the same way by all operations. This applies especially to valves. While no conflict arises if several operations need the same valve to be closed, no operation can demand it to be open at the same time.

More fine-grained models of resource sharing are possible. Sharing might depend, e.g., on the type of the operations that access a resource, or, simultaneous heating and filling are allowed, but parallel filling operations are not.

In simulation, different resource conflict handling strategies are possible:

- The most trivial possibility is to ignore them. All resources are considered to be sharable. This strategy is only meaningful if resource conflicts are not of interest for the simulation purpose or it is known in advance that no resource conflicts will occur. In this case, the absence of an expensive conflict detection mechanism can speed up the simulation.
- Another simple strategy is to detect conflicts but not to resolve them. If a conflict is detected, a warning can be issued or the simulation can be stopped. The task of reformulating the production plan or the recipe to avoid conflicts is thus given back to the user.
- A simple conflict resolution strategy is the first-come first-serve principle. If an operation attempts to access a resource which is not available, it must wait until the resource becomes available. If several operations are waiting, the order in which they can access the resource can be determined by the time of the request or by using a priority scheme. A sensible strategy may be to give operations of the recipe presently occupying the resource a higher priority than the requests of other recipes. This guarantees that a unit does not start processing another batch while the first batch is still in the unit.
- For batch processes, the notion of a "waiting" operation implies difficulties. Using the language of Petri nets, if an operation cannot start, the preceding transition cannot switch. This implies, however, that the preceding operation remains active. This does not make sense, e.g., if this operation is of type "filling" and the condition says "until Vessel C contains 5 m" [Engell et al., 1995]. In this case, a NOP action and an additional transition can be inserted. However, if the

© 2001 by CRC Press LLC

operation is "heat Vessel C until 380 K," it may be appropriate to insert an action to hold the temperature.

- More advanced conflict resolution strategies try to avoid conflicts by deferring the start of single operations or of whole recipes, or by changing the equipment assignment chosen in the control recipe. These strategies are usually out of the scope of a simulator, because they severely change the simulation model. Such strategies belong, rather, to the level of planning and scheduling tools.

6.11 Visualization

Simulation runs usually produce vast quantities of data describing the course of the simulation and the state of the model at different times. For a comprehensive analysis of the results, these data usually have to be processed, filtering out irrelevant information and compressing the relevant ones. Graphical representations are quite common since they enable the user to interpret the results at a glance.

A variety of different types of diagrams exist, each having advantages for different problems. Among these are:

- Trajectories that show the value of a certain quantity over time, e.g., temperatures or levels of vessels.
- Gantt charts to show the usage of equipment by the various recipes over time.
- Pie or beam charts representing the busy, idle, and waiting times of equipment or recipe steps.

Many software packages exist that are specialized in the numerical and/or statistical analysis of this data and its visualization. These packages expect the data to be in special data formats and most of them accept a variety of formats.

The approach in BASIP is not to provide a set of visualization components, but rather to offer interfaces to such packages (although the model editors can be used for online animation). Output components in BASIP, therefore, serve as adapters to transform the data produced by the simulation into a format that is understood by a particular graphics package. The interface may operate off-line, producing files in that format, or online, directly communicating with the programs if such an interface exists on their side.

The visualization packages should be usable independent of the simulator being employed. It would be a considerable effort to add new simulators or visualization packages if an adapter would be required for each combination of simulator and visualization packages. By the approach taken in BASIP, only a single adapter is required for each visualization package working with any simulator.

To make this approach work, each simulator has to use a standardized format for its output. This standardization is done using a protocol of so-called messages. For each piece of data a simulator produces, it sends a message to all the output clients. It does not need to know how many output clients there are, or of what type they are, thus, completely decoupling the simulation from the display of the results.

A message consists of a category, a predefined name, and message-specific data. Table 6.1 shows the existing categories and their meaning. Based on category and name, an output client can select the messages it is interested in and process the associated data.

TABLE 6.1 Message Categories in BASIP

Category	Meaning
Time	Current simulation time
Event	Event taking place
State	Current state of the model
Info	Other information about the simulation progress
Error	Errors in model execution

© 2001 by CRC Press LLC

So far, adapters to the software packages MATLAB and GNUPLOT exist. Other adapters serve as interfaces to the scheduling tool described in Section 6.13.

A special case of visualization is the online animation of the progress of the simulation. Animation gives the user a good imagination of what is happening inside the simulator and can provide a better understanding of the process.

For batch processes, animation means to display the current state of the recipe execution and of the plant, showing which equipment is currently in use.

It is straightforward to use the graphical editors of plants and recipes for this purpose. Animation is realized by output clients who set up inter-process communication connections to the graphical editors and send them the data needed for animation as it is generated. Hence, the editors are equipped with such communication facilities and can react to animation commands.

6.12 Example 1: A Simple Batch Process

The first example demonstrates the application of the different simulation strategies. The plant considered is a simple batch plant at the University of Dortmund that is used for research and teaching purposes. The plant flow sheet model, created with the BASIP plant editor, is shown in Figure 6.16.

The process consists of several steps. At first, highly concentrated saline solution is discharged from Vessel B1 to B3 where it is diluted with water from B2 until a given concentration is reached. Via buffer vessel B4, the contents are released into the evaporator B5. There, the solution is electrically heated and the water is evaporated until the salt concentration in the remaining solution has reached a certain level. The steam is condensed in a cooling coil and the condensation is collected in vessel B6. In the next step, the contents of B5 are discharged into B7 where they are cooled. At the same time, the water in B6 is cooled down. To save raw materials, both substances are pumped back into their respective storage vessels B1 and B2.

The recipe describing this process is shown in Figure 6.17 In addition to the main process described above, it contains an alternative route that is executed if the raw materials are not available. The alternative route executes some preparation steps to provide these materials for the next recipe to be started.

In the production plan for this example, this recipe is executed six times at different starting times. The plan is composed of three orders which contain two batches each. Each batch uses the same control recipe.

This model can be simulated with all simulators contained in BASIP. No changes of the model are necessary to use a different simulator. A comparison of the results reveals a good agreement between the different simulators. The simplifications taken in the discrete-event simulator do not affect the accuracy of the simulation in this case.

As an example, the temperature and mass trajectories of vessel B6 (which serves to catch the condensing vapor) are shown in Figure 6.18 for the hybrid (dashed) and the discrete-event simulator (solid). These plots have been creating using the interface to GNUPLOT.

A deliberately introduced difference is the consideration of heat losses in the plant model used for the hybrid simulator, which are ignored in the model for discrete-event simulation. This results in different peak temperatures the vessel reaches during the filling of the hot condensed steam and in a drop in temperature during phases of inactivity. Also, the mass curves are slightly shifted because the evaporation step takes somewhat longer in the hybrid simulation due to the heat losses. Without that difference in the model, the results are identical.

The exponential vs. linear rise in temperature, however, results from the limited number of data points generated by the discrete simulation. If more events would have occurred during that phase, more data points would have been generated, and the curve would approximate the real exponential course more closely.

The difference in computation time, however, is considerable. While the hybrid simulation takes about 70 CPU seconds, the discrete-event simulation is finished in less than 7 CPU seconds on a Sun Sparc 10 workstation (where most of the time is spent for i/o writing the simulation results).

However, it is not correct to draw the conclusion that the discrete-event approach is superior to the hybrid one. It is quite easy to construct examples where the simplifications in the discrete-event simulator lead to inaccurate or even wrong results.

© 2001 by CRC Press LLC

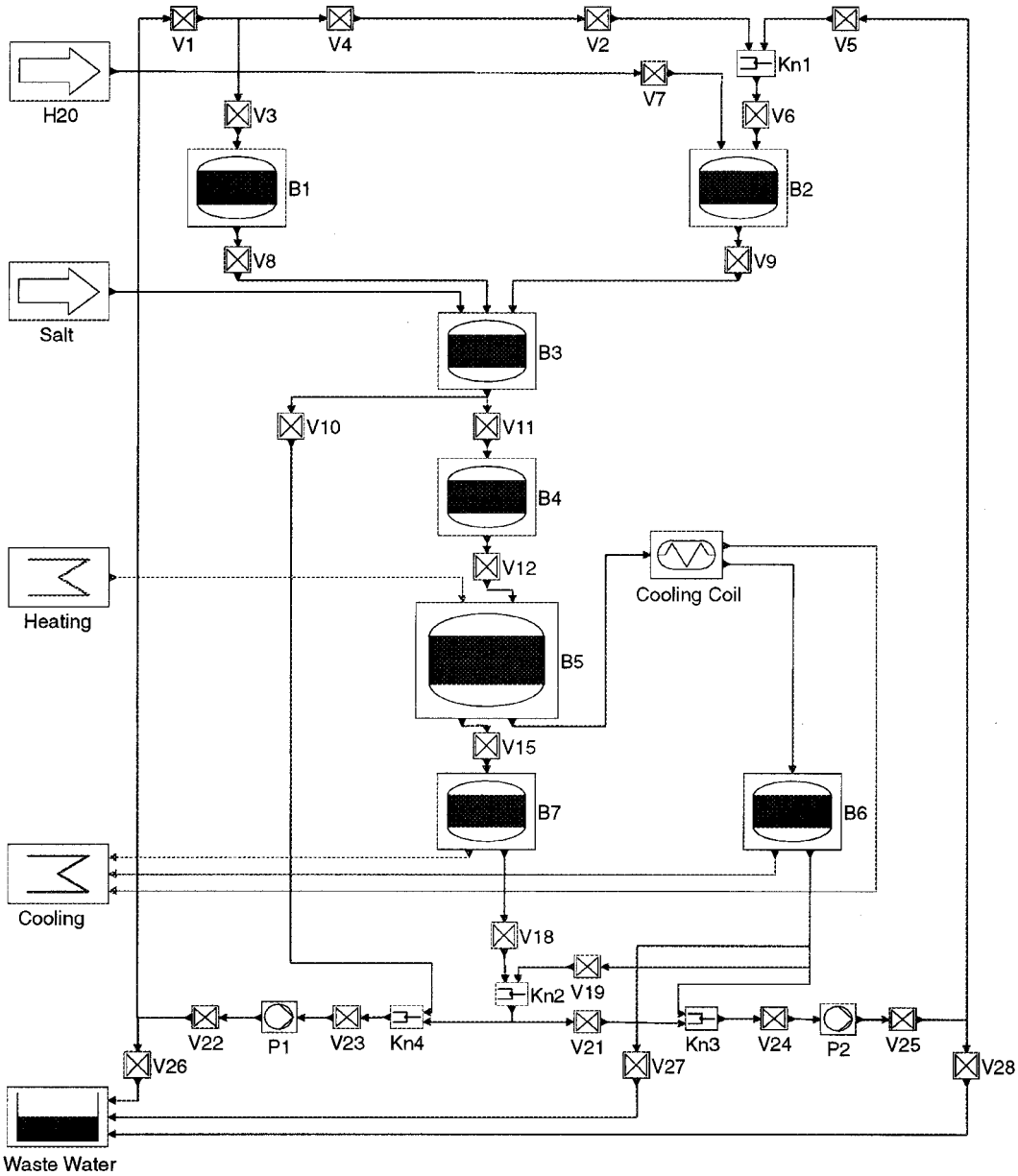


FIGURE 6.16 The example plant.

6.13 Example 2: Combining Scheduling and Simulation

This example shows the application of simulation in the context of production planning and scheduling. Production planning is known as a computationally hard problem, and the generation of optimal schedules for complex processes is considered almost impossible due to the numerous and often varying constraints that have to be taken into account. Most approaches therefore only try to produce good schedules that are robust to the inevitable variations in production conditions.

Figure 6.19 shows the architecture of a systemic for batch process scheduling called BASIS (batch simulation and scheduling) [Stobbe et al., 1997]. It consists of three main components.

© 2001 by CRC Press LLC

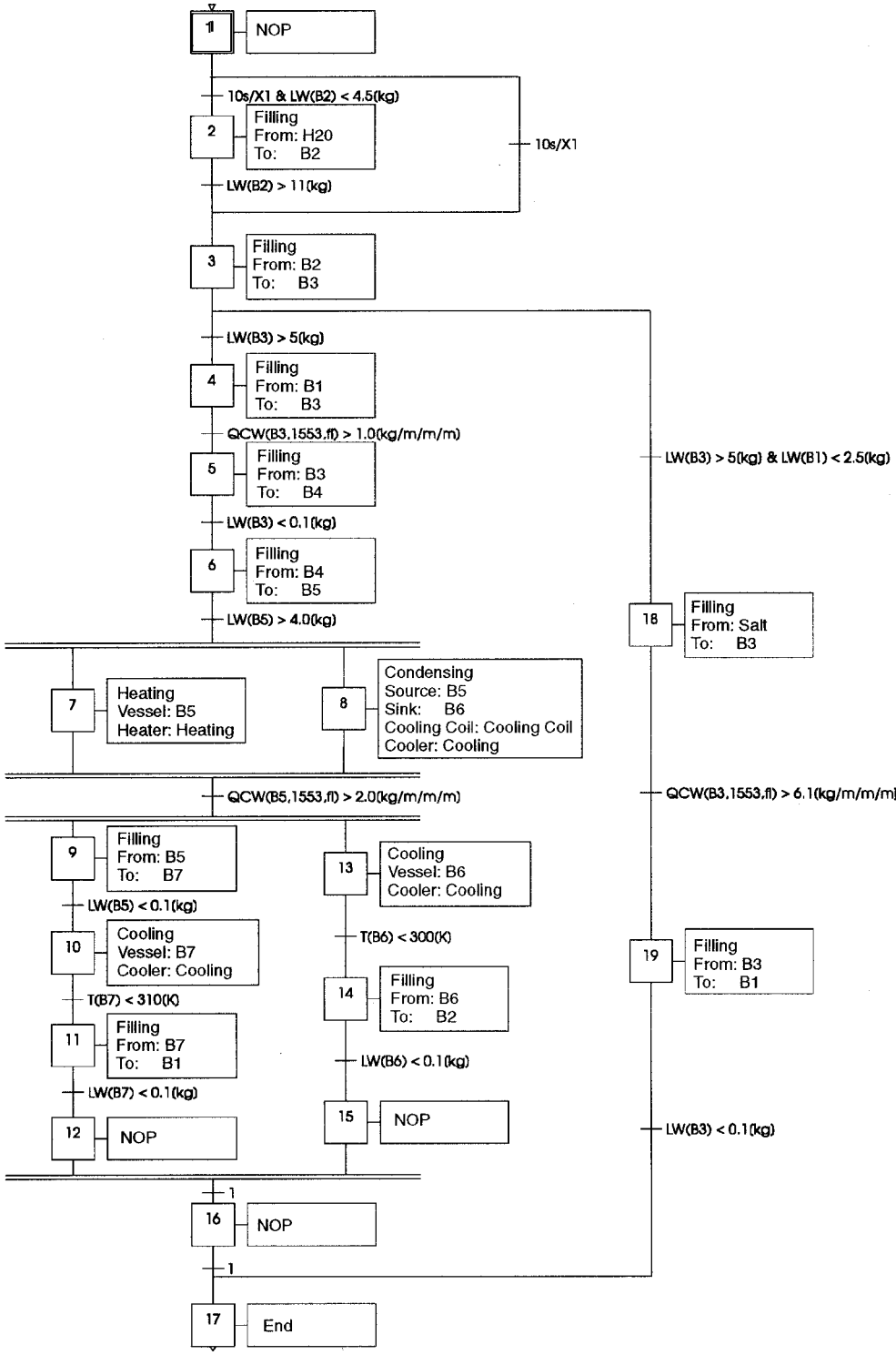


FIGURE 6.17 The master recipe for the example process.

© 2001 by CRC Press LLC

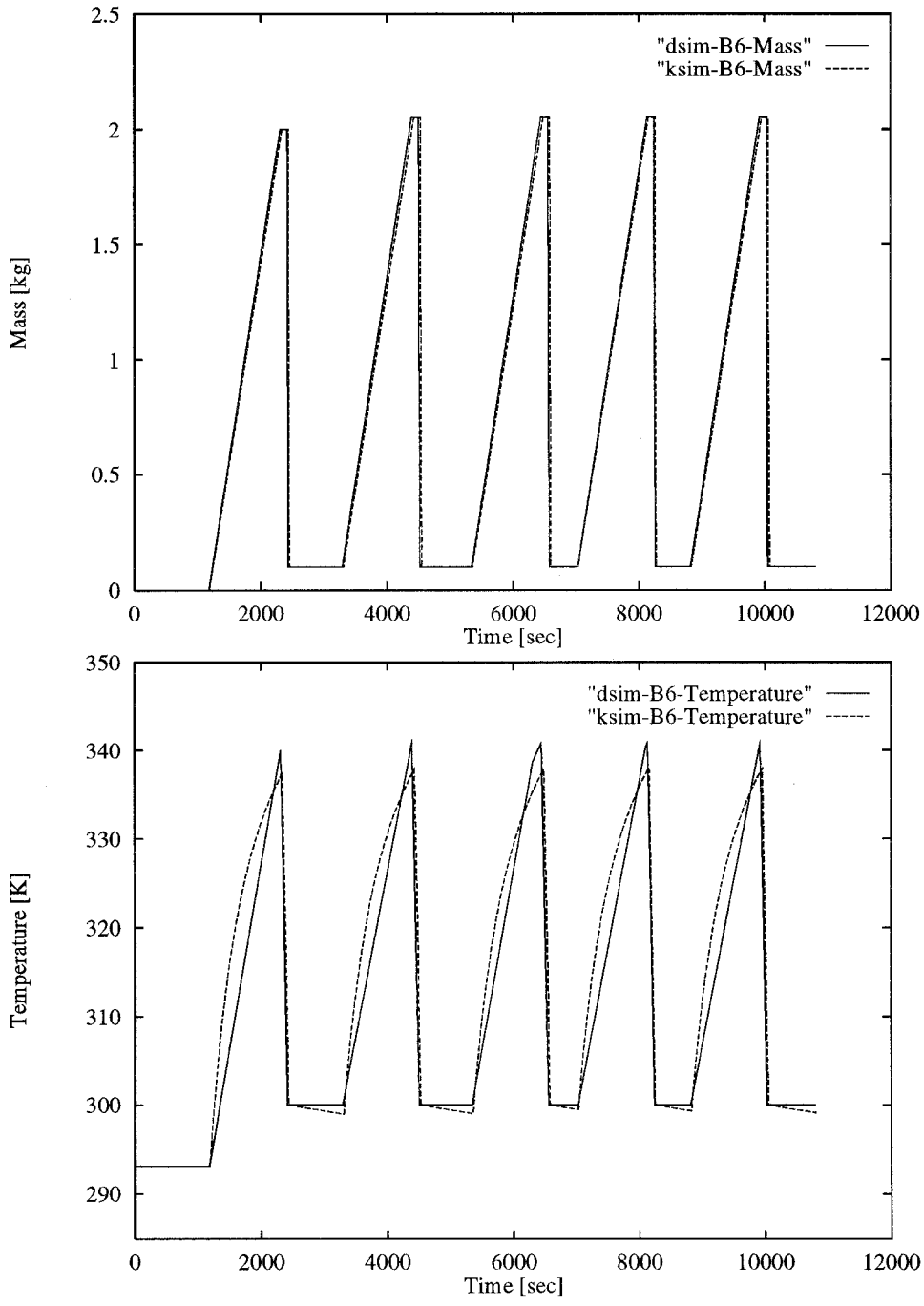


FIGURE 6.18 Trajectories for discrete and hybrid simulation.

The Leitstand represents the user interface for scheduling. Customer orders, consisting of a certain product, a quantity and a due date are maintained, and an initial production schedule for a configurable period of time is generated.

For that purpose, the Leitstand makes use of a schedule optimizer which creates a production plan by calculating the number of batches needed and determining the control recipe and starting time for

© 2001 by CRC Press LLC

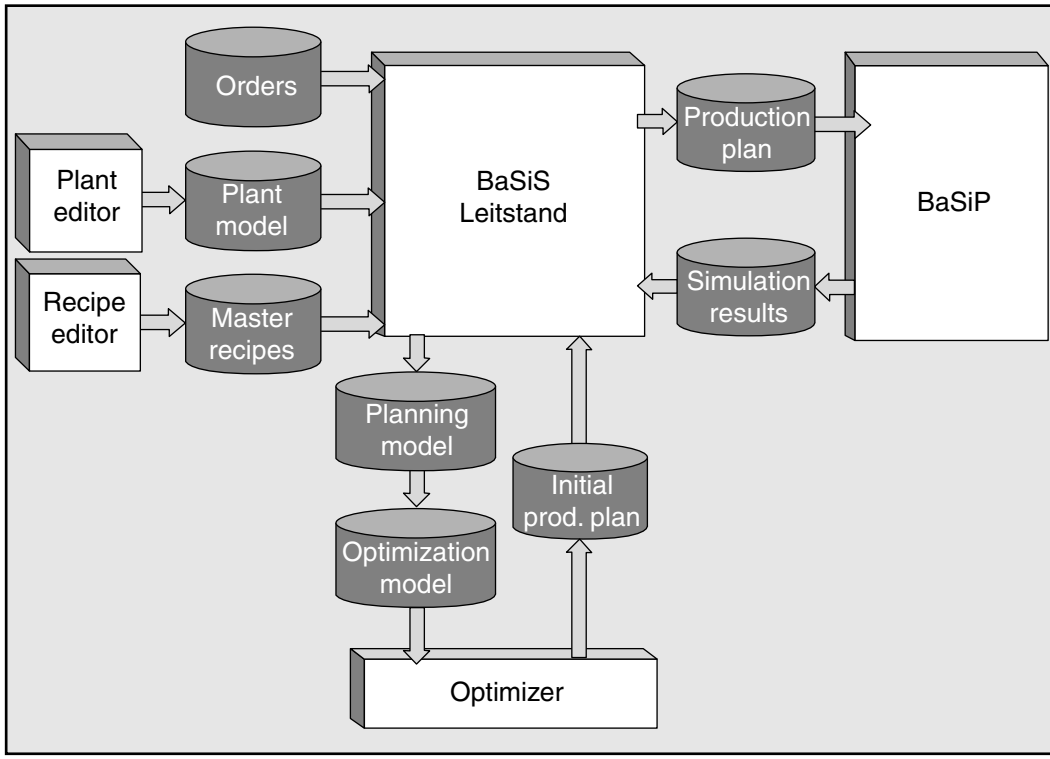


FIGURE 6.19 Architecture of the BASIS system.

each batch. It tries to optimize its decisions with respect to a given objective function, and, in this case, to minimize the difference between the demands and the actual production for each point of time. Positive and negative deviations can be weighted differently to punish late production more severely than production to stock. The optimization is performed using genetic algorithms or mathematical programming [Löhl, Schulz, and Engell, 1998].

To produce results in a sufficiently short time, the optimization algorithm uses a simplified model of the process that does not consider all the constraints. Because genetic algorithms perform an iterative improvement of their objective function, the optimization can be interrupted before its actual termination to guarantee fixed answering times of the optimizer. As a consequence, the plan generated by the optimizer most likely is not optimal and may be infeasible, i.e., violate some constraints that are not contained in the optimization model.

The more detailed model of the simulator enables a check of the resulting plan by simulation. A simulator can find violations of constraints and can verify the predicted execution times for recipes. The latter may vary due to waiting times in the recipe caused by resource conflicts.

Based on the comparison between the computed and the simulated schedule, the user can make improvements either by hand or by changing some bounds, e.g., the period of time for the schedule, and start the optimization again. After each step, simulation can be used for validation.

In the following, a short description of an example process is given by Schulz and Engell [1996]. The plant shown in Figure 6.20 is used to produce two different types of expandable polystyrene (EPS) in several grain fractions. The production process is divided into the main steps: preparation of raw material, polymerization, finishing of the polystyrene suspension in continuously operated production lines, and splitting into the different grain fractions for final storage. The process is of the flow shop type, i.e., all recipes have the same basic structure and differ only in the parameters and in certain steps.

© 2001 by CRC Press LLC

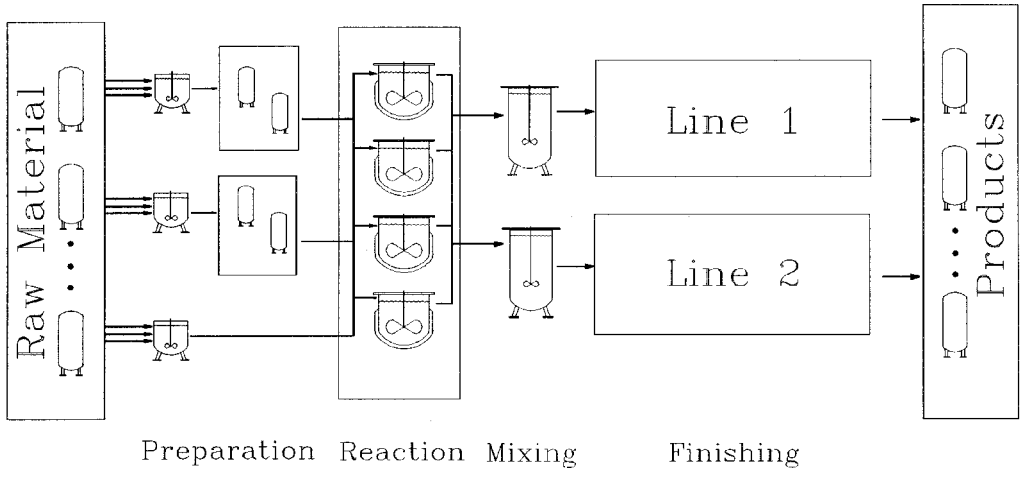


FIGURE 6.20 Batch plant for the production of EPS after [Schulz and Engell, 1996].

In the preparation step, batches of input material are mixed in vessels where they have to reside for a certain period ranging from half an hour to several hours. The mixtures are then pumped into one of several storage vessels, the capacity of which is adapted to the batch size in the polymerization step. There, further additives are added. The batches are only stable for a limited period and should not be discarded because of their toxicity, which makes it expensive to dispose of them. The storage vessels are grouped according to the types of polymer produced, and this assignment cannot be changed during plant operation. One of the inputs to the polymerization is a mixture of styrene and some additives. The number and the composition of the additives determine the grain size distribution of the product and the type of polystyrene. These parameters are not varied continuously, but certain fixed sets of parameters are used to control the grain size distribution.

To start the polymerization, input material is pumped into one of the reactors. The batch size is fixed in the polymerization step because the filling level of the reactors influences the grain size distribution. Variable batch sizes would introduce a complexity which currently cannot be handled appropriately on the technological level. The polymerization consists of four phases of almost equal duration.

For security restrictions, the start of a polymerization run in one reactor must be delayed for a certain time after the start in any other reactor.

The continuously operating finishing lines are coupled with the reactors by two vessels in which the batches are mixed. Each line is assigned to one type of polystyrene.

A major problem for production planning results from the limited influence of the free process parameters on the particle size distributions. The process parameters only affect the distribution among the fractions, but all fractions are always produced. Thus the production of a certain grain fraction cannot be related to a certain batch exclusively, and all batches are coupled.

The benefit of using simulation in this context is twofold. First, the feasibility of the generated plan can be verified. Violations of equipment boundaries, e.g., the maximum filling level of a vessel, and resource allocation conflicts are detected. Though the planning tool can check a number of constraints by heuristics itself, a complete simulation of the production schedule is the safest way to not miss violations of constraints.

The other benefit, even if the generated plan is feasible, is the detection of further optimization potential. In the case of the example process, two simplifications in the optimizer are leading to unwanted properties in the generated schedule.

The first is due to the time grid the optimizer is working with. The actual time of 16.4 h for the crucial process step, the polymerization, is rounded to 17 h. Consequently, in a two week planning horizon only 76 of the possible 79 polymerizations are planned. The simulation results show the resulting gaps in a

© 2001 by CRC Press LLC

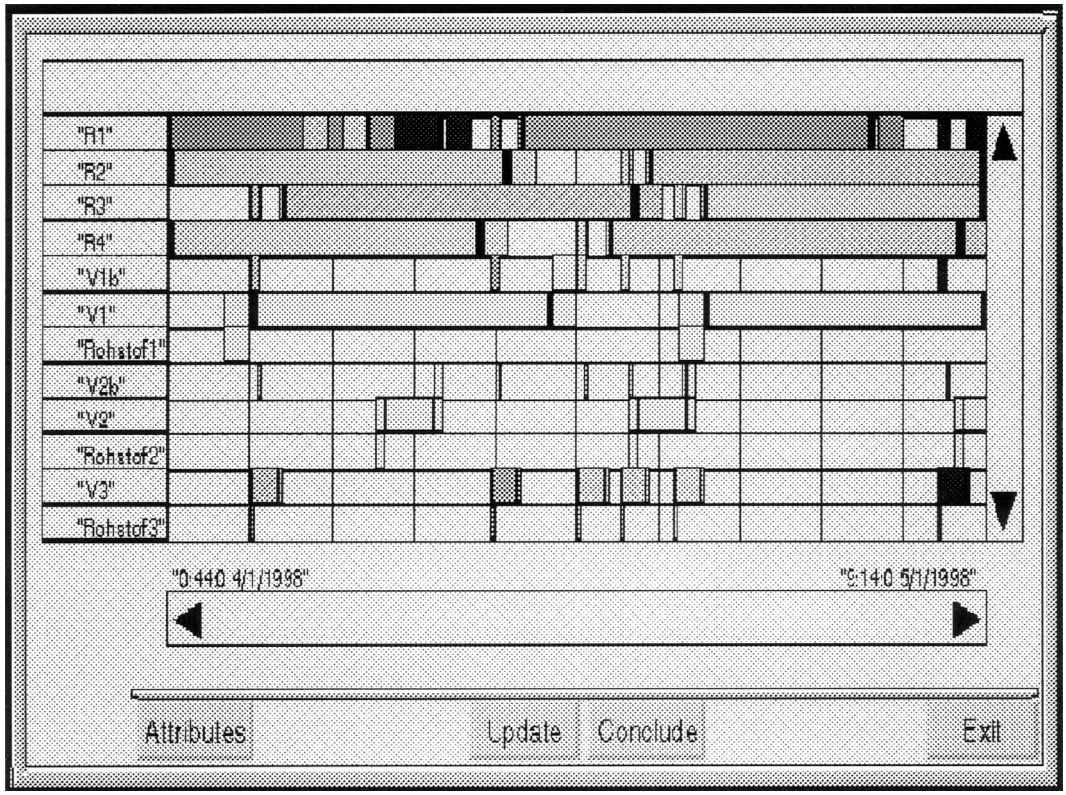


FIGURE 6.21 Simulation results in Gantt chart representation.

Gantt chart (see Figure 6.21). In this chart, the occupation of the plant equipment over time is shown. Different colors (greyscales) indicate different batches. A manual correction of the plan is possible by moving the beams in the chart.

The other simplification stems from the fact that the optimizer does not take the duration of a filling process into account, but rather assumes that a vessel is emptied immediately. This can lead to wrong predictions of the time when a product reaches the storage, and thus, the plan may fail to satisfy the customer demands on time, which is detected by the simulation as well.

In this application the discrete-event simulator is used, since the answering time of the simulation is crucial for an interactive application in which simulation is a background activity. Continuous simulation would not provide a gain of accuracy in this case. At the level of detail on which the process is viewed here, most process steps are trivial (mostly filling operations), while for the polymerization, no detailed model exists, and the resulting grain fractions are hard-coded in the recipes.

6.14 Discussion

Batch process control is a complex and multi-faceted task. Though the basic and intermediate control levels are well handled by modern DCS using the principle of recipe-driven production, computer support for high-level tasks in plant and production planning is still uncommon.

Simulation can give valuable insights into the dynamics of a system and thus provide support for decisions on higher levels. The hybrid nature of batch processes, however, requires special modeling and simulation methodologies to achieve satisfactory results. Either continuous or discrete-event approaches can be extended to meet the demands of this domain.

© 2001 by CRC Press LLC

The wide range of possible applications and the peculiarities that arise in batch process modeling put requirements on a modeling and simulation tool that no single monolithic approach can fulfill. Instead, a flexible and open system is needed that can be configured and extended to meet the requirements of a specific application in a specific environment.

The flexible architecture of BASIP is a prototype of such a system. It provides intuitive graphical modeling, simulation with the technique best suited for the specific purpose, and visualization with the user's favorite tools. However, some aspects in BASIP require further improvement.

- BASIP supports hierarchical modeling by the possibility to create super-blocks that contain other parts of the model. It would support model building better, however, to allow different semantics on different levels. This means that the modeling paradigm could be changed from level to level. Such multi-model approaches have been developed in theory, but are still quite rare in practice.
- The incorporation of specialized, or more detailed model blocks, is difficult. If they cannot be composed from blocks already existing in the BASIP model library, an extension is only possible at the source code level. A possibility to describe model blocks without access to the source code and without specific programming knowledge would be desirable.
- When simulation models become large, either due to the size of the plant or because of the time horizon to be simulated, the run times, even of a discrete-event simulation, may become infeasible for time-critical applications. The only resort here is to distribute the simulation over several processors. While parallel and distributed simulation has become a well established technique in recent years, very few efforts have been made so far to apply these techniques to hybrid systems. Much basic research still has to be done in this field.

6.15 Appendix: Object-Oriented Design

The object-oriented paradigm has had a fundamental impact on software development in the last decade. The software system presented here was designed and implemented entirely using the object-oriented approach. Therefore, the description of design and implementation aspects was given in terms of object-oriented design notations. In this appendix, a brief introduction into the fundamental concepts of object-orientation and the graphical notation of the design methodology used is given.

The general difference between object-oriented and conventional software construction is in how a software system is viewed. The classical structured approach views a program as a set of active procedures operating on passive data structures. In contrast, from an object-oriented point of view, a program is a collection of objects working together to achieve a common goal. While this definition seems to imply that conventional and object-oriented design are contradictory methodologies, object-orientation is in fact, a further development of the structured approach.

Though there is no commonly accepted definition of object-orientation, there is general agreement that the following characteristics should be included [Rumbaugh et al., 1993]:

1. Identity. Data is assigned to discrete, distinguishable entities—the objects. Each object has its own unique identity and can be accessed via a handle.
2. Classification. Objects with the same data structure and the same behavior can be grouped to classes. Classes are abstractions of object properties. An object is called an instance of its class.
3. Polymorphism. means that the same operation can have different behaviors in different classes. A specific implementations of an operation in a class is called a method. An operation can be performed without knowledge of which method is actually called for.
4. Inheritance. means a hierarchic relation between classes. A derived class inherits all properties of its base class. A derived class may add new or refine inherited properties. Inheritance can help to avoid redundant sections of code.

Object-orientation supports design and implementation principles that are valuable in any design paradigm. The most important of these are abstraction and encapsulation.

© 2001 by CRC Press LLC

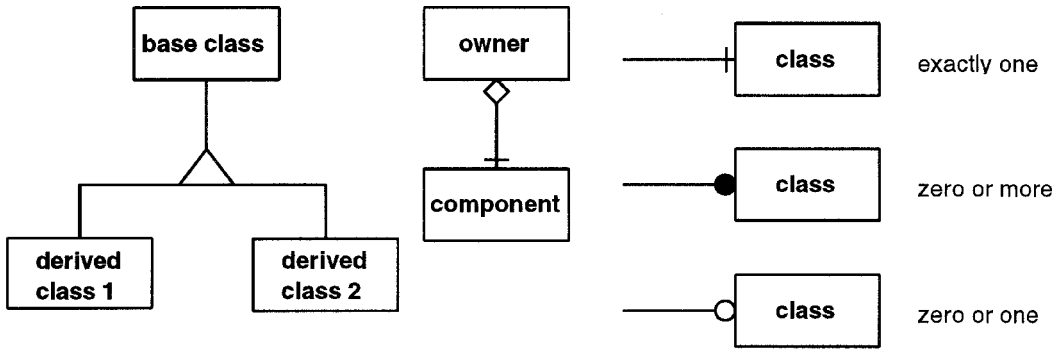


FIGURE 6.22 Notation for the OMT object model.

Another advantage of object-orientation is a common conceptual base for all phases of software development, especially analysis, design, and implementation. Several object-oriented methodologies exist which differ little in their concepts, but rather in the notation used. The methodology chosen here is among the most commonly used ones: the object modeling technique (OMT) after Rumbaugh et al. [1993]. With the release of the unified modeling language (UML) which merges OMT with two other popular approaches, there may develop a standard for the future [Fowler and Scott, 1997].

OMT uses three different models to describe a system, each using its own graphical notation. Here we use only the object model that describes the static structure of the system. The other two are the dynamical model, which focuses on the aspects of a system that change dynamically, and the functional model, which describes the data transformations of the system. The graphical elements of the object model notation used in this contribution are explained in Figure 6.22. Classes are depicted as boxes and have a unique name.

Relations between classes are called associations and are depicted as lines between classes. Special forms of associations are inheritance for which a triangle is used and aggregation, which means that an object is contained in another. Its symbol is a diamond. Associations have a multiplicity that expresses for how many objects of the given class the relation holds.

References

Barton, P.I. and Pantelides, C.C., 1994, Modeling of combined discrete/continuous processes, *AIChE 40,* 6, 996–979.

Engell, S., 1997, Modelling and analysis of hybrid systems, *Proc. 2nd MATH-MOD*, also to appear in a special issue of *IMACS-MCS,* pp. 17–32, Vienna.

Engell, S., Fritz, M., Schulz, C., and Wöllhaf, K., 1995, Hybrid simulation of flexible batch plants, *Proc. IFAC Symposium DYCORD+ '95*, pp. 123–128, Helsingør, Denmark.

Fowler, M. and Scott, K., 1997, *UML Distilled. Applying the Standard Object Modeling Language,* Addison-Wesley, Reading, MA.

Fritz, M. and Engell, S., 1997, An open software architecture for batch process simulation, *Comp. Chem. Eng.,* Proc. PSE '97/E'CAPE-7 21, 13, 769–773.

Fritz, M., Schulz, C., and Wöllhaf, K., 1995, A general graphic editor for complex simulation models, *Proc. EUROSIM' 95,* pp. 141–146, Vienna.

Genrich, H. J., Hanisch, H.-M., and Wöllhaf, K., 1994, Verification of recipe-based control procedures by means of predicate/transition nets, *15th International Conference on Application and Theory of Petri Nets,* Zaragoza, Spain.

Haxthausen, N., 1995, SP88 — the painkiller in validation, ISA Transactions 35, 369–378.

© 2001 by CRC Press LLC

International Electrotechnical Committee (IEC), 1988, *IEC 848: Preparation of Function Charts for Control Systems*.

The Instrumentation, Systems and Automation Society (ISA), 1994, *S88.01 Batch Control Part 1: Models and Terminology, Draft 12*, http://www.isa.org.

Loehl, T., Schulz, C., and Engell, S., 1998, Sequencing of batch operations for a highly coupled production process: genetic algorithms versus mathematical programming, *Comp. Chem. Eng.*, 22, S579-S585.

NAMUR-Arbeitskreis, 1992, 2.3. *NE 33: Aforderungen an Systeme zur Rezeptfahrweise (Requirements on Systems for Recipe Control)*, http://www.namur.de.

Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F., and Lorensen, W., 1993, *Object-Oriented Modeling and Design*, Prentice-Hall, Englewood Cliffs, NJ.

Schulz, C., and Engell, S., 1996, Mathematical modeling for on-line optimization of a multiproduct plant, *Proc. I-CIMRPO '96*, pp. 184–195, Eindloven, The Netherlands.

Stobbe, M., Fritz, M., Löhl, T., and Engell, S., 1997, BaSiS: Simulationsbasierte Produktionsplanung für rezeptgesteuerte Mehrproduktanlageu, *11. Symposium Simulationstechnik ASIM 97*, pp. 290–295, Dortmund.

Wöllhaf, K., 1995, *Objektorientiertf Modellierung and Simulation verfahrenstechnischer Mehrproduktanlagen (Object-oriented modelling and simulation of multipurpose batch plants),* Ph.D. thesis, Dept. of Chemical Engineering, University of Dortmund, Verlag Shaker, Aachen, Germany.

Wöllhaf, K., Fritz, M., Schulz, C., and Engell, S., 1996, BaSiP—batch process simulation with dynamically reconfigured process dynamics, *Comp. Chem. Eng.,* Proc. ESCAPE-6, 20, 972, 1281–1286.

© 2001 by CRC Press LLC

7

Scheduling Systems and Techniques in Flexible Manufacturing Systems

Ernest L. McDuffie
Florida State University

Marco Cristofari
Florida State University

Franco Caron
Politecnico di Milano

Massimo Tronci
University of Rome

William J. Wolfe
University of Colorado at Denver

Stephen E. Sorensen
Hughes Information Technology Systems

Preface
7.1 Flexible Manufacturing Systems and Performance Parameters
Background • Manufacturing Systems • Transfer Line and Job Shop: Characteristics and Limitations • Flexible Manufacturing System Technology • Flexibility
7.2 Scheduling Issues in FMS
Scheduling in FMS Technology • Performance Parameters • Static and Dynamic Scheduling • Static • Constructive Heuristics • Dynamic Approach • Research Trends in Dynamic Scheduling Simulation Approach • Experimental Approach for Simulated Dynamic Scheduling (ESDS) • Input Product-Mix and ESDS in a Random FMS • Data Analysis • Mix-Oriented Manufacturing Control (MOMC)
7.3 Summary

Preface

In this chapter a number of issues relating to scheduling in the flexible manufacturing system (FMS) domain will be discussed in detail. The chapter is divided into four main areas. First, performance parameters that are appropriate for a FMS will be covered. Here we will look at the background of manufacturing systems along with their characteristics and limitations. FMS technology and the issue of flexibility will also be looked at in this section. In the following section a detailed description of scheduling issues in an FMS will be presented. Here we will cover both static and dynamic scheduling along with a number of methods for dealing with these scheduling problems. The third major section of this chapter will detail a new approach to these issues called mix oriented manufacturing control (MOMC). Research trends and an experimental design approach for simulated dynamic scheduling (ESDS) will be covered in this section along with operational details and examples. Finally, the chapter will close with a summary of the information presented and conclusions.

© 2001 by CRC Press LLC

7.1 Flexible Manufacturing Systems and Performance Parameters

Background

In a competitive world market, a product must have high reliability, high standards, customized features, and low price. These challenging requirements have given a new impulse to all industrial departments and in particular, the production department. The need for flexibility has been temporarily satisfied at the assembly level. For example, several similar parts, differing from each other in few characteristics, e.g., color or other small attributes, are produced in great quantities using traditional manufacturing lines, and then are assembled together to produce smoothly different products. Unfortunately, this form of flexibility is unable to satisfy increasingly differentiated market demand. The life cycle of complex products, e.g., cars, motorbikes, etc., has decreased, and the ability to produce a greater range of different parts has become strategic industrial leverage. Manufacturing systems have been evolving from line manufacturing into job-shop manufacturing, arriving eventually at the most advanced expression of the manufacturing system: the FMS.

Manufacturing Systems

Based on flexibility and through-put considerations, the following manufacturing systems are identifiable

1. Line manufacturing (LM). This structure is formed by several different machines which process parts in rigid sequence into a final product. The main features are high through-put, low variability in the output product-mix (often, only one type of product is processed by a line), short flow time, low work-in-process (WIP), high machine utilization rate, uniform quality, high automation, high investments, high unreliability (risk of production-stops in case of a machine breakdown), and high time to market for developing a new product/process.
2. Job-shop manufacturing (JSM). The workload is characterized by different products concurrently flowing through the system. Each part requires a series of operations which are performed at different work stations according to the related process plan. Some work centers are interchangeable for some operations, even if costs and quality standards are slightly different from machine to machine. This feature greatly increases the flexibility of the system and, at the same time, the cost and quality variability in the resulting output. The manufacturing control system is responsible to choose the best option based on the status of the system. In a job-shop, the machines are generally grouped together by technological similarities. On one side, this type of process-oriented layout increases transportation costs due to the higher complexity of the material-handling control. On the other side, manufacturing costs decrease due to the possibility of sharing common resources for different parts. The main features of a job-shop are high flexibility, high variability in the output product-mix, medium/high flow time, high WIP, medium/low machine utilization rate, non-uniform quality, medium/low automation level, medium/low investments, high system reliability (low risk of production-stop in case of a machine breakdown), and for developing a new product/process.
3. Cell manufacturing. Following the criteria of group technology, some homogeneous families of parts may be manufactured by the same group of machines. A group of machines can be gathered to form a manufacturing cell. Thus, the manufacturing system can be split into several different cells, each dedicated to a product family. Material-handling cost decreases, while at the same time flexibility decreases and design cost increases. The main features of cell-manufacturing systems range between the two previously mentioned sets of system characteristics.

Transfer Line and Job Shop: Characteristics and Limitations

A comparison of the first two types of manufacturing systems listed, LM and JSM, can be summarized as follows. The main difference occurs in the production capability for the former and the technological capability for the latter. This is translated into high throughput for LM and high flexibility for JSM. A

© 2001 by CRC Press LLC

number of scheduling issues become apparent during different phases in these systems. These phases and different issues are:

1. Design phase. In case of LM, great care is to be taken during this phase. LM will operate according to its design features, therefore, if the speed of a machine is lower than the others, it will slow down the entire line, causing a bottleneck. Similar problems will occur in the case of machine breakdowns. Availability and maintenance policies for the machines should be taken into account during the design phase. Higher levels of automation generate further concern during the design phase because of the risk stemming from the high investment and specialization level, e.g., risk of obsoleteness. On the other hand, the JSM is characterized by medium/low initial investment, a modular structure that can be easily upgraded and presents less problems in the design phase. The product-mix that will be produced is generally not completely defined at start-up time, therefore, only the gross system capacity may be estimated on the basis of some assumptions about both processing and set-up time required. The use of simulation models highly improves this analysis.
2. Operating phase. The opposite occurs during the operating phase. In an LM, scheduling problems are solved during the design stage, whereas in a JSM, the complexity of the problem requires the utilization of a production activity control (PAC) system. A PAC manages the transformation of a shop order from the planned state to the completed state by allocating the available resources to the order. PAC governs the very short-term detailed planning, executing, and monitoring activities needed to control the flow of an order. This flow begins the moment an order is released by the planning system for execution, and terminates when the order is filled and its disposition completed. Additionally, a PAC is responsible for making a detailed and final allocation of labor, machine capacity, tools, and materials for all competing orders. Also, a PAC collects data on activities occurring on the shop floor involving order progress and status of resources and makes this information available to an upper level planning system. Finally, PAC is responsible for ensuring that the shop orders released to the shop floor by an upper level planning system, i.e., manufacturing requirement planning (MRP), are completed in a timely and cost effective manner. In fact, PAC determines and controls operation priorities, not order priorities. PAC is responsible for how the available resources are used, but it is not responsible for determining the available level of each resource. In short, PAC depends on an upper level planning system for answers to the following questions

- What products to build?
- How many of each product to build?
- When the products are due?

Scheduling in a job-shop is further complicated by the dynamic behavior of the system. The required output product-mix may change over time. Part types and proportion, deadlines, client requirements, raw material quality and arrival time, system status, breakdowns, bottlenecks, maintenance stops, etc., are all factors to be considered. A dynamic environment represents the typical environment in which a JSM operates. The complexity of the job-shop scheduling problem frequently leads to over-dimensioning of the system capacity and/or high levels of WIP. A machine utilization coefficient may range between 15–20% for nonrepetitive production. Some side effects of WIP are longer waiting time in queue and a manufacturing cycle efficiency (MCE) ranging from 1/25–1/30 for the job-shop, compared to approximately one for line manufacturing. MCE is defined as the ratio between the total processing time necessary for the manufacturing of a part and the flow time for that part, which is equal to the sum of total processing, waiting, setup, transporting, inspection, and control times.

Flexible Manufacturing System Technology

The computer numerical control (CNC) machine was the first stand-alone machine able to process several different operations on the same part without any operator's intervention. Integration of several machines moving toward an FMSs required the following steps:

© 2001 by CRC Press LLC

1. Creation of load/unload automatic device for parts and tools between storage and working positions.
2. Implementation of a software program in the central computer to control the tool-material load/unload automatic devices.
3. Automation of a parts-tools handling system among CNCs for all the cells.
4. Implementation of a software program in the central computer to control the parts-tools transportation system such as an automatic guided vehicle (AGV).
5. Implementation of a program-storage central computer, connected to each CNC, to automate the process of loading/unloading software programs necessary to control each manufacturing process.

The resulting FMS is a highly automated group of machines, logically organized, and controlled by a central computer (CC), and physically connected with an automated transport system. A CC schedules and provides data, software programs referring to the manufacturing processes to be performed, jobs, and tools to single workstations. Originally, FMS hierarchical-structure was centralized in nature. Typically, a CC was implemented on a mainframe because of the large number of tasks and needed response time. More recently, with the increasing power of mini and personal computers (PCs), the FMS's hierarchical-structure has become a more decentralized structure. The PCs of each CNC are connected to each other, forming a LAN system. This decentralization of functions across local workstations highly increases both the reliability of the system and, if a dynamic scheduling control is implemented, the flexibility of the system itself.

FMS, as an automated job-shop, can be considered as a natural development that originated either from job-shop technology with increased levels of automation, or manufacturing line technology with increased levels of flexibility. Because an FMS's ability to handle a great variety of products is still being researched, the majority of installed FMS's produce a finite number of specific families of products. An objective of an FMS is the simultaneous manufacturing of a mix of parts, and at the same time, to be flexible enough to sequentially process different mixes of parts without costly, time consuming changeover requirements between mixes.

FMS's brought managerial innovation from the perspective of machine setup times. Decreasing the tool changing times to negligible values, FMSs eliminate an important job-shop limitation. Because of the significant change in the ratio between working times and setup times, FMSs modify the profit region of highly automated systems.

The realization of an FMS is based on an integrated system design which differs from the conventional incremental job-shop approach that adds the machines to the system when needed. Integrated system design requires the dimensioning of all the system components such as machines, buffers, pallets, AGV, managerial/scheduling criteria, etc., in the design phase.

Main management leverages for a FMS are:

1. Physical configuration of the system. Number and capacity of the machines, system transportation characteristics, number of pallets, etc.
2. Scheduling policies. Loading, timing, sequencing, routing, dispatching, and priorities.

Physical configuration is a medium long-term leverage. The latter is a short-term leverage that allows a system to adapt to changes occurring in short-term.

Many different FMS configurations exist, and there is considerable confusion concerning the definition of particular types of FMSs [Liu and MacCarthy, 1996]. From a structural point of view the following types of FMSs can be identified:

1. FMS. A production system capable of producing a variety of part types which consists of CNC or NC machines connected by an automated material handling system. The operation of the whole system is under computer control.
2. Single flexible machine (SFM). A computer controlled production unit which consists of a single CNC or NC machine with tool changing capability, a material handling device, and a part storage buffer.

© 2001 by CRC Press LLC

3. Flexible manufacturing cell (FMC). A type of FMS consisting of a group of SFMs sharing one common handling device.
4. Multi-machine flexible manufacturing system (MMFMS). A type of FMS consisting of a number of SFMs connected by an automated material handling system which includes two or more material handling devices, or is otherwise capable of visiting two or more machines at a time.
5. Multi-cell flexible manufacturing system (MCFMS). A type of FMS consisting of a number of FMCs, and possibly a number of SFMs, if necessary, all connected by an automated material handling system.

From a functional point of view, the following types of FMSs can be identified:

1. Engineered. Common at the very early stage of FMS development, it was built to process the same set of parts for its entire life cycle.
2. Sequential. This type of FMS can be considered a flexible manufacturing line. It is structured to sequentially manufacture batches of different products. The layout is organized as a flow-shop.
3. Dedicated. The dedicated FMS manufactures the same simultaneous mix of products for an extended period of time. The layout is generally organized as a flow-shop, where each type of job possibly skips one or more machines in accordance with the processing plan.
4. Random. This type of FMS provides the maximum flexibility, manufacturing at any time, any random simultaneous mix of products belonging to a given product range. The layout is organized as a job-shop.
5. Modular. In this FMS, the central computer and the transportation system are so sophisticated that the user can modify the FMS in one of the previous structures according to the problem at hand.

Flexibility

One of the main features of an FMS is its flexibility. This term, however, is frequently used with scarce attention to current research studies, whereas its characteristics and limits should be clearly defined according to the type of FMS being considered. The following four dimensions of flexibility — always measured in terms of time and costs — can be identified:

1. Long-term flexibility (years/months). A system's ability to react, at low cost and in short time, to managerial requests for manufacturing a new product not considered in the original product-set considered during the design stage.
2. Medium-term flexibility (months/weeks). Ability of the manufacturing system to react to managerial requests for modifying an old product.
3. Short-term flexibility (days/shifts). Ability of the manufacturing system to react to scheduling changes, i.e., deadlines, product-mix, derived from upper level planning.
4. Instantaneous flexibility (hours/minutes). Ability of the system to react to instantaneous events affecting system status, such as bottlenecks, machine breakdowns, maintenance stops, etc. If alternative routings are available for a simultaneous product-mix flowing through the system, a workload balance is possible concerning the different machines available in the system.

It should be noted that production changes generally do not affect the production volume, but instead, the product-mix. In turn, system through-put appears to be a function of product-mix. The maximum system through-put can be expressed as the maximum expected system output per unit time under defined conditions. It is important to identify the conditions or states of a system, and the relation between these states and the corresponding system through-put. These relations are particularly important for the product-mix. In general, each product passes through different machines and interact with each other. System through-put is a function of both the product-mix and any other system parameters, e.g., control policies, transportation speed, queue lengths etc., that may influence system performance. In conclusion,

© 2001 by CRC Press LLC

system through-put measures can not be defined as *a priori* because of the high variability of product-mix. Instead, system through-put should be evaluated, i.e., through simulation, for each considered product-mix.

7.2 Scheduling Issues in FMS

Considering the high level of automation and cost involved in the development of an FMS, all development phases of this technology are important in order to achieve the maximum utilization and benefits related to this type of system. However, because the main topic of this chapter is smart scheduling in an FMS, a vast available scientific literature in related areas, i.e., economical considerations, comparisons between FMS and standard technologies, design of an FMS, etc. are only referred to. These main development phases can be identified as (a) strategic analysis and economic justification, (b) facility design to accomplish long term managerial objectives, (c) intermediate range planning, and (d) dynamic operations scheduling.

In multi-stage production systems, scheduling activity takes place after the development of both the master production schedule (MPS) and the material requirements planning (MRP). The goal of these two steps is the translation of product requests (defined as requests for producing a certain amount of products at a specific time) into product orders. A product order is defined as the decision-set that must be accomplished on the shop floor by the different available resources to transform requests into products.

In the scheduling process the following phases can be distinguished as (a) allocation of operations to the available resources, (b) allocation of operations for each resource to the scheduling periods, and (c) job sequencing on each machine, for each scheduling period, considering the job-characteristics, shop floor characteristics, and scheduling goals (due dates, utilization rates, through-put, etc.)

In an FMS, dynamic scheduling, as opposed to advance sequencing of operations, is usually implemented. This approach implies making routing decisions for a part incrementally, i.e., progressively, as the part completes its operations one after another. In other words, the next machine for a part at any stage is chosen only when its current operation is nearly completed. In the same way, a dispatching approach provides a solution for selecting from a queue the job to be transferred to an empty machine.

It has been reported that the sequencing approach is more efficient than the dispatching approach in a static environment; however, a rigorous sequencing approach is not appropriate in a dynamic manufacturing environment, since unanticipated events like small machine breakdowns can at once modify the system status.

The complexity of the scheduling problem arises from a number of factors:

1. Large amount of data, jobs, processes, constraints, etc.
2. Tendency of data to change over time.
3. General uncertainty in such items as process, setup, and transportation times.
4. The system is affected by events difficult to forecast, e.g., breakdowns.
5. The goals of a good production schedule often change and conflict with each other.

Recent trends toward lead time and WIP reduction have increased interest in scheduling methodologies. Such an interest also springs from the need to fully utilize the high productivity and flexibility of expensive FMSs. Furthermore, the high level of automation supports information intensive solutions for the scheduling problem based both on integrated information systems and on-line monitoring systems. Knowledge of machine status and product advancement is necessary to dynamically elaborate and manage scheduling process.

Before coping with the scheduling problem, some considerations must be made (a) actual FMSs are numerous, different, and complex. The characteristics of the observed system must be clearly defined, (b) the different types of products produced by a system can not be grouped together during a scheduling

© 2001 by CRC Press LLC

problem, and (c) decisions on what, how much, and how to produce, are made at the upper planning level. PAC activity can not change these decisions.

The following assumptions are generally accepted in scheduling (a) available resources are known, (b) jobs are defined as a sequence of operations or directed graph, (c) when the PAC dispatches a job into the shop floor, that job must be completed, (d) a machine can not process more than one job at a time, (e) a job can not be processed contemporary on more than one machine, and (f) because the scheduling period is short, the stock costs are discarded.

Scheduling in FMS Technology

Flexibility is a major consideration in the design of manufacturing systems. FMSs have been developed over the last two decades to help the manufacturing industry move towards the goal of greater flexibility. An FMS combines high levels of flexibility, high maximum throughput, and low levels of work-in-progress inventory. This type of system may also allow unsupervised production. In order to achieve these desirable benefits, the control system must be capable of exercising intelligent supervisory management. Scheduling is at the heart of the control system of a FMS. The development of an effective and efficient FMS scheduling system remains an important and active research area.

Unlike traditional scheduling research, however, a common language of communication for FMS scheduling has not been properly defined. The definition of a number of terms relevant to FMS scheduling are as follows [Liu and MacCarthy, 1996]:

Operation. The processing of a part on a machine over a continuous time period.

Job. The collection of all operations needed to complete the processing of a part.

Scheduling. The process encompassing all the decisions related to the allocation of resources to operations over the time domain.

Dispatching. The process or decision of determining the next operation for a resource when the resource becomes free and the next destination of a part when its current operation has finished.

Queuing. The process or decision of determining the next operation for a resource when the resource becomes free.

Routing. The process or decision of determining the next destination of a part when its current operation has finished.

Sequencing. The decision determining the order in which the operations are performed on machines.

Machine set-up. The process or decision of assigning tools on a machine to perform the next operation(s) in case of initial machine set-up or tool change required to accommodate a different operation from the previous one.

Tool changing. It has a similar meaning to machine set-up, but often implies the change from one working state to another, rather than from an initial state of the machine.

System set-up. The process or decision of allocating tools to machines before the start of a production period with the assumption that the loaded tools will stay on the machine during the production period.

All of the above concepts are concerned with two types of decisions (a) assignment of tools, and (b) allocation of operations. These two decisions are interdependent. Loading considers both. Machine set-up or tool changing concentrates on the tool assignment decisions made before the start of the production period or during this period, assuming the allocation of operations is known in advance. Dispatching concentrates on the operation allocation decisions, leaving the tool changing decision to be made later, or assuming tools are already assigned to the machines.

Mathematicians, engineers, and production managers have been interested in developing efficient factory operational/control procedures since the beginning of the industrial revolution. Simple decision rules can alter the system output by 30% or more [Barash, 1976]. Unfortunately, the results of these studies are highly

© 2001 by CRC Press LLC

dependent on manufacturing system details. Even relatively simple single-machine problems are often NP-hard [Garey and Johnson, 1979] and, thus, computationally intractable. The difficulty in solving operational/control problems in job-shop environments is further compounded by the dynamic nature of the environment. Jobs arrive at the system dynamically over time, the times of their arrivals are difficult to predict, machines are subject to failures, and managerial requests change over time. The scheduling problem in an FMS is similar to the one in job-shop technology, particularly in case of the random type. Random FMSs are exposed to sources of random and dynamic perturbations such as machine breakdowns, changes over time in workload, product-mix, due dates, etc. Therefore, a dynamic and probabilistic scheduling approach is strongly advised. Among the different variable sources, changes in WIP play a critical role. This system parameter is linked to all the other major output performance variables, i.e., average flow time, work center utilization coefficients, due dates, setup total time dependence upon job sequences on the machine, etc.

Random FMS scheduling differs from job-shop scheduling in the following specific features and are important to note before developing new scheduling methodologies [Rachamadugu and Stecke, 1994]:

1. Machine set-up time. System programmability, robots, automatic pallets, numerical controlled AGVs, etc., decrease the machine set-up times to negligible values for the different operations performed in an FMS. The main effect of this characteristic is the ability to change the manufacturing approach from batch production to single item production. This has the benefit of allowing each single item the ability to choose its own route according to the different available alternative machines and system status. In a job-shop, because of the large set up time required for each operation, the production is usually performed in batch. Due to the contemporary presence of several batches in the system, the amount of WIP in a job-shop is generally higher than in an analogous FMS.
2. Machine processing time. Due to the high level of automation in an FMS, the machine processing times and the set up times can often be considered deterministic in nature, except for randomly occurring failures . In a job-shop, due to the direct labor required both during the set up and the operation process, the set up and processing times must be considered random in nature.
3. Transportation time. In a job-shop, due to large batches and high storage capacity, the transportation time can be dismissed if it is less than the average waiting time in a queue. In an FMS, because of low WIP values, the transportation time must be generally considered to evaluate the overall system behavior. The AGV speed can be an important FMS process parameters, particularly in those cases in which the available storage facilities are of low capacity.
4. Buffer, pallet, fixture capacity. In an FMS, the material handling and storage facilities are automated, and therefore specifically built for the characteristics of the handled materials. For this reason the material handling facilities in an FMS are more expensive than in a job-shop, and economic constraints limit the actual number of facilities available. Therefore, physical constraints of WIP and queue lengths must be considered in FMS scheduling, whereas these constraints can be relaxed in job-shop scheduling.
5. Transportation capacity. AGVs and transportation facilities must be generally considered as a restricted resource in an FMS due to the low volume of WIP and storage facilities that generally characterized these systems.
6. Instantaneous flexibility — alternative routing. The high level of computerized control typical in an FMS makes available a large amount of real-time data on the status of the system, which in turn allows for timely decision making on a best routing option to occur. In a job-shop, routing flexibility is theoretically available, but actually difficult to implement due to the lack of timely information on the system status.

Performance Parameters

The scheduling process refers to the assignment and timing of operations on machines on the shop floor. Scheduling goals are generally difficult to formulate into a well defined objective function that focuses on costs for the following reasons:

© 2001 by CRC Press LLC

1. Translation of system parameters, i.e., machine utilization coefficient, delivery delays, etc., whose values are important to the evaluation of the fitness of a schedule into cost parameters is problematic.
2. Significant changes in some parameters, such as stock level, in the short-term bring only a small difference to the cost evaluation of a schedule.
3. Some factors derived from medium-term planning can have a great influence on scheduling costs, but can not be modified in short-term programming.
4. Conflicting targets must be considered simultaneously.

Because scheduling alternatives must be compared on the basis of conflicting performance parameters, a main goal is identified as a parameter to be either minimized or maximized, while the remaining parameters are formulated as constraints. For instance, the number of delayed jobs represents an objective function to be minimized under the constraint that all jobs meet their due dates.

The job attributes generally given as inputs to a scheduling problem are (a) processing time, t_{jik} where j = job, i = machine, k = operation, (b) possible starting date for job j, s_j, and (c) due date for job j, d_j.

Possible scheduling output variables that can be defined are (a) job entry time, E_j, (b) job completion time, C_j, (c) lateness, $L_j = C_j - d_j$, (d) tardiness, $T_j = \max(0, L_j)$, and (e) flow time, $F_j = C_j - E_j$.

Flow time represents a fraction of the total lead time. When an order is received, all the management procedures that allow the order to be processed on the shop floor are activated. Between the date the order is received and the possible starting date s_j, a period of time which is equal to the sum of procedural time and waiting time passes (both being difficult to quantify due to uncontrollable external elements, e.g., new order arrival time). At time s_j the job is ready to enter the shop-floor and therefore belongs to a set of jobs that can be scheduled by PAC. However, time s_j and time E_j do not necessarily coincide because PAC can determine the optimal order release to optimize system performance. Flow time is equal to the sum of the processing time on each machine, included in the process plan for the considered part, and the waiting time in queues. The waiting time in queue depends on both the set-up time and the interference of jobs competing for the same machine. In the short-term, waiting time can be reduced because both of its components depend on good scheduling, whereas the processing time can not be lessened.

Given a set of N jobs, the scheduling performance parameters are:

Average lateness

$$LM = \frac{\sum_{j=1}^{N} L_j}{N} \tag{7.1}$$

Average tardiness

$$TM = \frac{\sum_{j=1}^{N} T_j}{N} \tag{7.2}$$

Average flow time

$$FM = \frac{\sum_{j=1}^{N} F_j}{N} \tag{7.3}$$

© 2001 by CRC Press LLC

Number of job delays

$$ND = \sum_{j=1}^{N} \delta(T_j) \tag{7.4}$$

where,

$$\delta(T_j) = 1 \text{ se } T_j > 0; \text{ and } \delta(T_j) = 0 \text{ se } T_j = 0.$$

Makespan

$$MAK = \max_j\{C_j\} - \min\{E_j\} \tag{7.5}$$

Machine i utilization coefficient

$$TU_i = \frac{\sum_{j=1}^{N} t_{ij}}{MAK} \tag{7.6}$$

where, t_{ji} = processing time of job j at machine i.
Average system utilization coefficient

$$TUM = \frac{\sum_{i=1}^{M}\sum_{j=1}^{M} t_{ji}}{MAK \qquad M} \tag{7.7}$$

where M = number of machines in the system.
Work-in-progress

$$WIP = \frac{1}{b-a}\int_a^b WIP(t)dt \tag{7.8}$$

where, $WIP(t)$ = number of jobs in the system at time t; $a = \min_j (E_j)$; and $b = \max_j (C_j)$
Total set-up time

$$SUT = \sum_{j=1}^{M} SU_i \tag{7.9}$$

where, SU_i = set-up time on machine i to process the assigned set of jobs.

© 2001 by CRC Press LLC

Set-up times depend on system and job characteristics:

1. Set-up time can be independent from the job sequence. In this case, it is useful to include the set-up time in the processing time t_{ji}, nullifying the corresponding Su_i.
2. Set-up time can be dependent on two consecutive jobs occurring on the same machine; therefore, a set-up time matrix is employed.
3. Set-up time can be dependent on the history which depends on the job sequence on that machine.

Typical scheduling goals are: (a) minimization of average lateness, (b) minimization of average tardiness, (c) minimization of average flow time, (d) minimization of number of job delays, (e) minimization of makespan, (f) maximization of average system utilization coefficient, (g) minimization of WIP, and (h) minimization of total setup time.

Generally, one objective is not always preferable over the others, because production context and priorities derived from the adopted production strategy determine the scheduling objective. If jobs do not have the same importance, a weighted average for the corresponding jobs can be applied instead of a single parameter, i.e., average lateness, average tardiness, or average flow time.

Besides the average flow time, its standard deviation can be observed. Similar considerations can be made for the other parameters. In several cases the scheduling goal could be to minimize the maximum values of system performance variables, for instance, the lateness or tardiness for the most delayed job, rather than minimizing the average values of system performance variables.

Economical value of WIP can sometimes be a more accurate quantifier of the immobilized capital inside a production system. Because the time interval to be scheduled is short, the importance of decisions based on stock costs is generally low. Therefore, the minimization of WIP is commonly not used. High capacity utilization, which is often overvalued, gradually loses weight against low inventories, short lead times, and high due date performance.

Static and Dynamic Scheduling

Almost all types of scheduling can be divided into two large families, static scheduling and dynamic scheduling.

Static Scheduling

Static scheduling problems consist of a fixed set of jobs to be run. Typically, the static scheduling approach (SSA) assumes that the entire set of jobs arrive simultaneously, that is, an entire set of parts are available before operations start, and all work centers are available at that time [Vollman, Berry, and Whybark, 1992]. The resulting schedule is a list of operations that will be performed by each machine on each part in a specific time interval. This static list respects system constraints and optimizes (optimization methodologies) or sub-optimizes (heuristic methodologies) the objective function. The performance variable generally considered as an objective function to be minimized is the makespan, defined as the total time to process all the jobs. SSA research deals with both deterministic and stochastic processing/set-up times. The general restrictions on the basic SSA can be summarized as follows [Hadavi et al., 1990; Mattfeld, 1996].

1. The tasks to be performed are well defined and completely known; resources and facilities are entirely specified.
2. No over-work, part-time work, or subcontract work is considered.
3. Each machine is continuously available; no breakdown or preventive maintenance.
4. There is only one machine of each type on the shop floor; no routing problem.
5. Each operation can be performed by only one machine on the shop floor; no multipurpose machine.
6. There is no preemption; each job must be processed to completion, if started.
7. Each machine can process only one job at a time.
8. Jobs are strictly-ordered sequences of operations without assembly operations.

© 2001 by CRC Press LLC

9. No rework is possible.
10. No job is processed twice on the same machine.
11. The transportation vehicle is immediately available when the machine finishes an operation; AGVs are not considered to be resources.
12. Jobs may be started at any time; no release time exists.

Some of the assumptions mentioned above can be seen in some job-shops, but they are generally not valid in many real cases. Job-shop and random FMS are characterized by an environment highly dynamic in nature. Jobs can be released at random. Machines are randomly subject to failures and managerial requests change over time. The changing operating environment necessitates dynamic updating of scheduling rules in order to maintain system performance. Some of the above mentioned conditions are relaxed in some optimization/heuristic approaches, but still the major limitation of these approaches is that they are static in perspective. All the information for a scheduling period is known in advance and no changes are allowed during the scheduling period. In most optimization/heuristic approaches, processing/set-up/transportation times are fixed and no stochastic events occur. These methodologies are also deterministic in nature.

Dynamic Scheduling

Dynamic scheduling problems are characterized by jobs continually entering the system. The parts are not assumed to be available before the starting time and are assumed to enter the system at random, obeying for instance a known stochastic inter-arrival time distribution. Processing and set-up times can be deterministic or stochastic. No fixed schedule is obtained before the actual operations start. Preemptions and machine breakdowns are possible, and the related events are driven by a known stochastic time variable. The standard tool used to study how a system responds to different possible distributed dispatching policies is the computer simulation approach. Flexibility of a system should allow a part to choose the best path among different machines, depending on the state of the system at a given time. This is often not achieved in reality due to the difficulty in determining the best choice among a large number of possible routing alternatives.

Short-sighted heuristic methods are often used because they permit decomposing one large mathematical model that may be very difficult to solve, into a number of smaller problems that are usually much easier to solve. This approach looks for a local optimal solution related to a sub-problem. It assumes that once all the smaller problems have been optimized, the user is left with an optimal solution for the large problem. Short-sighted heuristic methods generally are strongly specific to the problem to be solved. This approach is currently popular because it allows the use of more realistic models, without too many simplifications, simultaneously keeping computation time within reasonable limits.

Two other important areas of research are the prediction of task duration for future scheduling when little historical information is available, and the representation of temporal relationships among scheduled tasks. Both of these areas are important in that they deal with time at a fundamental level. How time is represented is critical to any scheduling methodology. One concept that has been proposed call smart scheduling deals with these areas by implementing Allen-type temporal relation in a program that allows the scheduling of these types of complex relationships [McDuffie et al., 1995]. An extension of this concept also deals with the problem of predicting future task duration intervals in the face of little historical data [McDuffie et al., 1996]. This extension uses ideas from fuzzy logic to deal with the lack of data.

Static

Manual Approach

Traditionally, manual approaches are most commonly used to schedule jobs. The principle method type in this area is the backwards interval method (BIM). BIM generally works backwards from the due dates of the jobs. The operation due date for the last operation must be the same as the order due date. From the order due date, BIM schedules jobs backwards through the routing. BIM subtracts from the operation due date the time required by all the activities to be accomplished. These activities

© 2001 by CRC Press LLC

usually include the operation time, considering the lot size and the setup time; the queue time, based on standard historical data; and the transportation time. After the operation lead time has been identified, the next due date for the preceding operation can be determined. The procedure is repeated backwards through all the operations required by the order. This approach is used to define MRP. BIM's major limitation is that both the system through-put and lead times are assumed to be known, generally evaluated by historical data. Furthermore, its complications grow when more than one order is considered. More details can be found in [Fox and Smith, 1984], [Smith, 1987], and [Melnyk and Carter, 1987].

Mathematical Approaches

In the 1960s, [Balas, 1965, 1967] and [Gomory, 1965, 1967] utilized the growing power of the computer to develop modern integer programming. This methodology allows some type of job-shop scheduling problems to be solved exactly. A large number of scheduling integer programming formulations and procedures can be found in [Greenberg, 1968; Florian, Trepant, and MacMahon, 1971; Balas, 1970; Schwimer, 1972; Ignall and Schrage, 1965; McMahon and Burton, 1967]. Once a suitable function for each job has been determined, there are two main optimal algorithms that can be applied [Wolfe, Sorensen, and McSwain, 1997]:

- Depth First. A depth first-search is carried out to exhaustively search all possible schedules and find the optimal schedule. This search is computationally expensive. For this reason and others, this method has been replaced by the branch-and-bound method and dynamic programming.
- Branch-and-Bound. With this method, it is possible to prune many branches of the search tree.

Even if the mathematical approach allows the user to find the optimal solution restricted by the given constraints, its use is limited to simple case studies. Consider the simple case of 30 jobs to be scheduled on a single machine. The mathematical method will search the solution space, which can be viewed as a tree structure whose first level has 30 branches, for the possible choices for the first job. Then, for the second level there will again be 29 choices for each branch, giving $30*29 = 870$ choices or second level branches. Therefore, the complete solution-space tree will consist of $30*29*(28)* \ ... \ *1 = 30! = 2 * 10^{32}$ branches at the 30th level. The depth first method looks at all branches, evaluates, and compares them. Using the branch-and-bound method, parts of the tree are pruned when it is determined that only a non-optimal solution can be in that branch [Morton and Pentico, 1993].

In the 1960s and 1970s, dynamic programming was used for sequencing problems. Such methods are competitive with branch-and-bound, particularly for some restricted class of problems. For both integer and dynamic programming, small problems can be solved and optimal solutions found. Large, realistic problems however, have remained, and in all likelihood will continue to be intractable [Morton and Pentico, 1993]. The rough limit is of 50 jobs on 1 machine [Garey et al., 1976].

Constructive Heuristics

Priority Dispatch (PD) and Look Ahead (LA) Algorithms

The priority dispatch and the look ahead algorithms are constructive algorithms: they build a schedule from scratch. They use rules for selecting jobs and for allocating resources to the jobs. They can also be used to add to an existing schedule, usually treating the existing commitments as hard constraints. They are fast algorithms primarily because there is no backtracking. They are easy to understand because they use three simple phases: selection, allocation, and optimization. The LA method is similar to the PD approach but uses a much more intelligent allocation step. The LA algorithm looks ahead in the job queue, i.e., considers the unscheduled jobs, and tries to place the current job to cause the least conflict with the remaining jobs. See [Syswerda and Palmucci, 1991], [Wolfe, 1994, 1995a, 1995b], and [Wolfe, Sorensen, and McSwain, 1997] for a detailed study of these algorithms. We begin by describing the PD algorithm.

© 2001 by CRC Press LLC

This method starts with a list of unscheduled jobs and uses three phases: selection, allocation, and optimization to construct a schedule. There are many possible variations on these steps, and here we present a representative version.

Phase I: Selection. Rank the contending, unscheduled, jobs according to:

$$\text{rank}(\text{job}_i) = \text{priority}_i$$

We assume that a priority value is given to each job. It is usually a function of the parameters of the job: due date, release time, cost, profit, etc. This is just one way to rank the jobs. Other job features could be used, such as duration or slack. Slack is a measure of the number of scheduling options available for a given job. Jobs with high slack have a large number of ways to be scheduled, and conversely, jobs with low slack have very few ways to be scheduled. Sorting by priority and then breaking the ties with slack (low slack is ranked higher) is an excellent sorting strategy. This is consistent with the established result that the WSPT algorithm is optimal for certain machine sequencing problems. It, too, uses a combination of priority and slack. In this case, the slack is measured by the inverse of the duration of the job and the queue is sorted by:

$$\text{rank} = \text{priority/duration}$$

Phase II: Allocation. Take the job at the top of the queue from Phase I and consider allocating the necessary resources. There are usually many ways to allocate the resources within the constraints of the problem. Quite often a mixed strategy is most effective. For example, allocate the least resources required to do the job at the best times. This gets the job on the schedule but also anticipates the need to leave room for other jobs as we move through the queue. Continue through the job queue until all jobs have had one chance to be scheduled.

Phase III: Optimization. After Phases I and II are complete, examine the scheduled jobs in priority order and increase the resource allocations if possible, e.g., if a job can use more time on a resource and there are no conflicts with scheduled jobs or other constraint violations, allocate more time to the job. This is a form of hill climbing; in fact, any hill climbing algorithm could be plugged in at this step, but the theme of the PD is to keep it simple.

The advantages of the PD approach are that it is simple, fast, and produces acceptable schedules most of the time. The simplicity supports the addition of many variations that may prove useful for a specific application. It also makes the PD easy to understand so that operators and users can act in a consistent manner when quick changes are necessary. The PD produces optimal solutions in the simple cases, i.e., when there is little conflict between the jobs. The accuracy of the PD, however, can be poor for certain sets of competing jobs. If the PD does not provide good solutions we recommend the look ahead method, which sacrifices some speed and simplicity for more accuracy.

Look Ahead Algorithm (LA): A powerful modification can be made to the PD method at the allocation step, when the job is placed at the best times. This can be greatly improved by redefining "best" by looking ahead in the queue of unscheduled jobs. For each possible configuration of the job, apply the PD to the remaining jobs in the queue and compute a schedule score (based on the objective function) for the resulting schedule; then, unschedule all those jobs and move to the next configuration. Schedule the job at the configuration that scores the highest and move to the next job in the queue, with no backtracking. This allows the placement of a job to be sensitive to down-stream jobs. The result is a dramatic improvement in schedule quality over the PD approach, but an increase in run time. If the run times are too long, the depth of the look ahead can be shortened. The LA algorithm is an excellent trade off between speed and accuracy. The LA algorithm is near optimal for small job sets, and for larger job sets it usually outperforms the PD by a significant margin. The LA algorithm is difficult to beat and makes an excellent comparison for modern algorithms such as genetic and neural methods.

© 2001 by CRC Press LLC

Improvement Heuristics

Improvement heuristic (IH) is also called neighborhood search. They first consider a feasible starting solution, generated by any method, then try ways to change a schedule slightly, i.e., slide right/left, grow right/left, shrink right/left. These changes stay in the neighborhood of the current solution and an evaluation for each resulting schedule is produced. If there is no way to get an improvement, the method is finished. Otherwise, IH takes the result with the largest improvement and begins looking for small changes from that. IHs are a particular application of the general programming method called hill climbing. The following methodologies are particular applications of IH:

- Simulated annealing (SA). Adds a random component to the classical IH search by making a random decision in the beginning, a gradually less random decision as it progresses, and finally converges to a deterministic method. Starting with the highest priority job, the simplest method of SA randomly picks one of the moves and considers the difference in the objective function: ΔQ = score after move-score before move. If $\Delta Q > 0$, then SA makes the move and iterates. If $\Delta Q \geq 0$, then SA randomly decides to take or not take the move [Kirkpatrick, Gelatt, and Vecchi, 1983; Van Laarhoven, Aarts, and Lenstra, 1992; Caroyer and Liu, 1991; Ishibuchi, Tamura, and Tanaka, 1991].
- Genetic algorithms (GA). Refers to a search process that simulates the natural evolutionary process. Consider a feasible population of possible solutions to a scheduling problem. Then, in each generation, have the best solutions be allowed to produce new solutions — children, by mixing features of the parents or by mutation. The worst children die off to keep the population stable and the process repeats. GA can be viewed as a broadened version of neighborhood search [Della, Tadei, and Volta, 1992; Nakano and Yamada, 1991; Davis, 1991; Dorigo, 1989; Falkenaur and Bouffoix, 1991; Holland, 1975].
- Tabu search (TS). Is a neighborhood search with an active list of the recent moves done. These moves are tabu. Every other move, except ones on the list, are possible and the results are compared. The best result is saved on the list, then, the best updated solution found is saved on another list; therefore, when a local optimum is reached, the procedure will move on to a worse position in the next move. The tabu list will prevent the algorithm from returning immediately to the last explored area which contains the found local optimum solution. In this manner, the algorithm is forced to move toward new unexplored solutions more quickly [Glover and Laguna, 1989; Laguna, Barnes, and Glover, 1989; Widmer and Herts, 1989; Glover, 1990].

Beam Search (BS)

One of a number of methods developed by AI for partially searching decision trees [Lowerre, 1976; Rubin, 1978; Ow and Morton, 1988], BS is similar to branch-and-bound, but instead of pruning a part of the tree that is guaranteed useless, BS cuts off parts of the tree that are likely to be useless. A major characteristics of the methodology is the determination of what "likely" means. By avoiding these parts of the solution space, search time is saved without taking much risk. An example of BS is the approximate dynamic programming method.

Bottleneck Methods (BM)

This method [Vollman, 1986; Lundrigan, 1986; Meleton, 1986; Fry, Cox, and Blackstone, 1992] is representative of other current finite scheduling systems such as Q-Control and NUMETRIX. Both programs are capable of finding clever approximate solutions to scheduling models. Briefly, BM identifies the bottleneck resource and then sets priorities on jobs in the rest of the system in an attempt to control/reduce the bottleneck; for instance, jobs that require extensive processing by a non-critical machine in the system, but use only small amounts of bottleneck time, are expedited. When considering only single bottleneck resource problems, using BM simplifies the problem so that it can be solved more easily. The main disadvantages of BM are [Morton and Pentico, 1993]; it can not deal with multiple and shifting bottle-

© 2001 by CRC Press LLC

necks, the user interface is not strong, reactive correction to the schedule requires full rerunning, and the software is rigid and simple modifications of schedules are difficult.

Bottleneck Dynamics (BD)

This method generates estimates of activity, prices for delaying each possible activity, and resource prices for delaying each resources on the shop floor. These prices are defined as follows [Morton and Pentico, 1993]:

- Activity prices. To simplify the problem, suppose each job in the shop has a due date and that customer dissatisfaction will increase with the lateness and inherent importance of the job. The objective function to be minimized is defined as the sum of dissatisfaction of the various customer for the jobs. Every estimated lead time to finish time for each activity yields a current expected lateness. If this activity is delayed for the time t^*, then the increase of lateness, and thus the increase of customer dissatisfaction, can be evaluated. This gives an estimate of the activity price or delay cost. If the marginal cost of customer is constant, the lead time is not relevant because the price is constant independent from the lateness of the job. The more general case coincides with the case in which the client is unsatisfied for the tardiness, but does not care if the job is early.
- Resources prices. If the resource is shut down at time t^* for a period p, all jobs in the queue are delayed by a period p; therefore, the resource delay cost will be at least the sum of all the prices of activities in the queue times the period p. At the same time, in the period p other jobs will arrive in the queue of the shut down machine. During time $(t^* + \Delta t)$, the price of the resource must be evaluated, as well as the prices of these activities during time $(p + \Delta t)$. All activities occurring during the busy period for that resource will be slowed down. This evaluation is clearly not easy, but, fortunately for the methodology, experimental results show that the accuracy of BD is not particularly sensitive to how accurate the prices are.

Once these evaluations are calculated, it is possible to trade off the costs of delaying the activities versus the costs of using the resources. This allows for the determination of which activity has the highest benefit/cost ratio so that it will be selected next from the queue to be scheduled. A dispatching problem can be solved in this way. One approach to the routing problem, in which each job has a choice of routes through a shop, is simply to calculate the total resource cost for each route (sum of the resource price times activity duration) as the jobs are routed in each path. The expected lateness cost must also be calculated and added to each path; then the path with the lowest cost can be chosen.

The main limitations of the methodology are in assessing the lead time of an activity, activity price, and assessing the number of entities in a queue, which is used to determine resource price. These values are dependent on the flow of jobs in the job-shop, which in turn depends on shop sequencing decisions that have not been made yet [Morton and Pentico, 1993]. The loop can only be broken by using approximation techniques to estimate the lead time of each activity:

1. Lead time iteration, i.e., running the entire simulation several times in order to improve the estimates.
2. Fitting historical or simulation data of lead times versus shop load measures.
3. Human judgment based on historical experience.

For more information see [Vepsalainen and Morton, 1988; Morton et al., 1988; Morton and Pentico, 1993; Kumar and Morton, 1991].

Dynamic Approach

Dispatching Rules

Dispatching is one of the most important topics in short-term production scheduling and control. Dispatching rules originated as a manual approach that considered a plethora of different priority rules,

© 2001 by CRC Press LLC

most of which being forward-looking in nature. The reason for considering such a large number of rules is that the determination of an optimum rule is extremely difficult. Using the computer simulation approach, several research studies have observed different system behaviors when various rules were simulated. Currently, distributed dispatching rules are utilized by applying AI technology to the problem of real-time dynamic scheduling for a random FMS.

The standard use of dispatching rules occurs when a PAC individualizes a set of dispatching rules for each decision point (routing, queuing, timing). These rules are then applied to the waiting orders, causing them to be ranked in terms of the applied rule. A dispatching rule makes decisions on the actual destination of a job, or the actual operation to be performed in real-time — when the job or the machine is actually available — and not ahead-of-time, as occurs in other types of scheduling approach. For this reason, dispatching rules ensure dynamic scheduling of an observed system. Dispatching rules must be transparent, meaningful, and particularly consistent with the objectives of the planning system [Pritsker, 1986; Graves, 1981; Montazeri and Van Wassenhove, 1990].

Based on job priority, queuing rules choose a job to be processed by a machine from its jobs waiting queue. Priority is the key factor in distinguishing the goodness of the chosen type of queuing rule. Queuing differs from sequencing in that queuing picks the job with the maximum priority, whereas, sequencing orders all the jobs in the queue.

An implicit limitation of the dispatching approach is that it is not possible to assess the completion time for each job due to the absence of preliminary programming extended to all the jobs to be scheduled. A commonly accepted advantage of the dispatching approach is its simplicity and low cost, due to both the absence of a complete preliminary programming and system adaptability to unforeseen events. Dispatching rules can be used to obtain good solutions for the scheduling problem. To compare their performance, simulation experiments can be implemented on a computer.

Dispatching rule classifications can be obtained on the basis of the following factors:

1. Processing or set-up time (routing rules).

Short processing time (SPT) or shortest imminent operation (SIO). The job with the shortest processing time is processed first.

Longest processing time (LPT). The job with the longest processing time is processed first.

Truncated SPT (TSPT). SPT is valid unless a job waits in the queue longer than a predetermined period of time, in which case this job is processed first, following FIFO rule.

Least work remaining (LWKR). The job with the minimum remaining total processing time is processed first.

Total work (TWORK). The job with the minimum total work time is processed first. The total work time is defined as the sum of the total processing time, waiting time in queue, and setup time.

Minimum set-up time (MSUT). The job with minimum setup time, depending on the machine status, is processed first.

2. Due date.

Earliest due date (EDD). The job with the earliest due date is processed first.

Operation due date (OPNDD). The job with the earliest operation due date is processed first. The operation due date is derived from the division of the time-interval within job delivery due date d_j and job entry time I_j, into a number of intervals equal to the number of operations in which the endpoint of each interval is the operation due date.

3. Processing time and due date.

Minimum slack time (MST). The job with the minimum slack time is processed first. Slack time is obtained by subtracting the current time and the total processing time from the due date.

Slack per operation (S/OPN). The job with the minimum ratio, which is equal to slack time, divided by the number of remaining operations is processed first.

© 2001 by CRC Press LLC

SPT with expediting (SPTEX). Jobs are divided into two classes of priority: jobs whose due dates can be subject to anticipation, or jobs with a difference between the due date and current time equal to zero or less, constitute the first priority class. All other jobs belong to the second type of class. The first priority class is scheduled before the second priority class, and first priority class follows SPT rule.

Slack index (SI). The value of SI is calculated as the difference between slack time and a control parameter equal for all the jobs. The control parameter considers the system delay. Jobs are divided into two classes based on the value of SI. The first class where $SI < 0$ is processed prior to the second class where $SI \geq 0$; in both cases SPT is used.

Critical ratio (CR). The job with minimum ratio, which is equal to the difference between the due date and current date divided by the remaining processing time, is processed first.

4. System status — balancing workload on machines.

CYC. Cyclic priority transfer to first available queue, starting from the last queue that was selected.

RAN. Random priority assigns an equal probability to each queue that has an entity in it.

SAV. Priority is given to the queue that has had the smallest average number of entities in it to date.

SWF. Priority is given to the queue for which the waiting time of its first entity is the shortest.

SNQ. Priority is given to the queue that has the current smallest number of entities in the queue.

LRC. Priority is given to the queue that has the largest remaining unused capacity.

ASM. In assembly mode option, all incoming queues must contribute one entity before a process may begin service.

Work in next queue (WINQ). The job whose next operation is performed by the machine with the lowest total work time is processed first.

5. Job status.

First in first out (FIFO). The job that enters the queue first is processed first.

Last in first out (LIFO). The job that enters the queue last is processed first.

First in the system first served (FISFS). The job that enters the system first is processed first.

Fewest remaining operations (FROP). The job with the fewest remaining operations in the system is processed first.

Most remaining operations (MROP). The job with the most remaining operations in the system is processed first.

6. Economical factors.

COVERT. The job with the highest ratio, which is equal to delay costs divided by remaining time, is processed first.

7. Combined rules.

- SPT/LWKR. The job with the minimum value for S is processed first. The control parameter is calculated as:

$$S = a * TO + (1 - a) * TR \tag{7.10}$$

where, TO is operation execution time, TR is remaining processing time, and a is weight constant.

Another classification criterion is based on the distinction between static and dynamic rules. A rule is static if the values input to the functional mechanism of the rule, independent from system status, do not change with time. The contrary is defined as dynamic. For instance, TWORK and EDD are static, whereas, S/OPN and CR are dynamic.

A third classification criterion is based on the type of information the rules utilize. A rule is considered local only if information related to the closest machine is utilized, while a global rule utilizes

© 2001 by CRC Press LLC

data related to different machines as well. For instance, SPT and MSUT are local, whereas, NINQ and WINQ are global.

Some characteristics of the rules can be outlined as follows:

- Some rules can provide optimal solutions for a single machine. The average flow time and the average lateness are minimized by SPT. Maximum lateness and tardiness are minimized by EDD.
- Rule performance is not generally influenced by the system dimension.
- An advantage of SPT is its low sensitivity to the randomness of the processing time.
- SPTEX performance is worse than SPT for average flow time, but better for average tardiness. If the system is overloaded, SPT rules seem to be more efficient than SPTEX, also for the average tardiness.
- EDD and S/OPN offer optimal solutions, whereas, SPT provides good solutions to satisfy due dates.

Computer Simulation Approach (CSA)

After the 1950s, the growth of computers made it possible to represent shop floor structure in high detail. The natural development of this new capability resulted in the use of computer simulation to evaluate system performance. Using different types of scheduling methodologies under different types of system conditions and environment, such as dynamic, static, probabilistic and deterministic, computers are able to conduct a wide range of simulations [Pai and McRoberts, 1971; Barret and Marman, 1986; Baker, 1974; Glaser and Hottenstein, 1982]. Simulation is the most versatile and flexible approach. The use of a simulator makes possible the analysis of the behavior of the system under transient and steady state conditions, and the *a priori* evaluation of effects caused by a change in hardware, lay-out, or operational policies.

In the increasingly competitive world of manufacturing, CSA has been accepted as a very powerful tool for planning, design, and control of complex production systems. At the same time simulation models can provide answers to "what if" questions only. CSA makes it possible to assess the values for output variables for a given set of input variables only.

Load-Oriented Manufacturing Control (LOMC)

Recent studies show how load oriented manufacturing control (LOMC) [Bechte, 1988] allows for significant improvement in manufacturing system performance. LOMC accomplishes this improvement by keeping the actual system lead times close to the planned ones, maintaining WIP at a low level, and obtaining satisfactory system utilization coefficients.

This type of control represents a robustness oriented approach, aiming at guaranteeing foreseeable, short lead times that provide coherence between the planning upper level parameters, e.g., MRP parameters, and the scheduling short-term parameters, and allowing for a more reliable planning process. If the manufacturing system causes a significant difference between planned and actual lead times, then the main consequence is that job due dates generally are not met.

LOMC is a heuristic and probabilistic approach. Simulation runs have shown that the probabilistic approach, which is inferior to the deterministic approach for a static and deterministic environment, appears to be the most robust in a highly dynamic environment, e.g., job-shop, random FMS.

According to LOMC, lead-time control is based on the following consideration: the average lead time at a work center equals the ratio of average inventory in the input buffer and average through-put. Therefore, if one wants average lead time to meet a given level, the ratio of inventory to through-put should be adjusted accordingly. If one wants a low dispersion of lead times, one must apply the FIFO rule while dispatching. The basic relationship between average lead time and average inventory to through-put ratio leads to the principle of load limiting. Load limits can be expressed for each work center as a percentage of the capacity of that work center.

The LOMC approach allows for the control of WIP in a system. System performance parameters, such as flow time, are generally correlated to WIP by assigning a specific load limit to each work center. Overall

© 2001 by CRC Press LLC

flow time in a system can be kept constant and short for many jobs without significantly decreasing the work center utilization coefficient. This result is obtained with an order release policy based on the following assumptions:

- A set of jobs (defined at the upper planning level, i.e., by MRP) are available for system loading and are allowed to enter the system when the right conditions (according to the adopted job release policy) are met.
- Jobs are released *if* and *only if* no system workload constraints are violated by the order release process; therefore, on-line system workload monitoring is required.
- Order release occurs in accordance with a given frequency based on a timing policy.
- In general, the workload is measurable in terms of work center machine hours, or utilization rate of the capacity available during the scheduling period.
- For downstream work centers, it is not possible to determine the incoming workload in an accurate way. A statistical estimation is performed, taking into account the distance in terms of number of operations to be performed on the job, of the work center from the input point of the FMS.

LOMC approach is based on the following phases:

1. The number of jobs to be completed for a given due date is determined at the upper planning level (MRP). Taking into account the process plan for each job, the workload for each work center is evaluated. In this phase, the medium-term system capacity can be estimated, and, at the same time, possible system bottlenecks can be identified. Due dates and lead times are the building blocks of the planning phase. A crucial point is the capability of the system to hold the actual lead times equal to the planned lead times.
2. Job release policy depends on its effects on WIP, lead times, and due dates. Order release procedure determines the number of jobs concurrently in the system and, therefore, the product-mix that balances the workload on the different work centers considering the available capacities. An order release procedure that does not consider the workloads leads to underloading or overloading of work centers.
3. Job dispatching policy manages the input buffers at the work centers. There are different available dispatching rules that should operate in accordance with the priority level assigned to the jobs at the upper planning level; however, underloading or overloading of work centers derived from a non-adequate job release procedure can not be eliminated by dispatching rules. With balanced workloads among the different work centers, many dispatching rules loose their importance, and simple rules like FIFO give an actual lead time close to the planned lead time.

The application of LOMC requires an accurate evaluation of a series of parameters:

1. Check time. Frequency of system status evaluation and loading procedure activation.
2. Workload aggregation level. Workload can be computed at different aggregation levels. On one extreme, the system overall workload is assessed. On the other extreme, workload for each work center is assessed. The latter is a more complex and efficient approach since it allows for the control of workload distribution among the work centers, avoiding possible system bottlenecks. An intermediate approach controls the workload only for the critical work centers that represent system bottlenecks. If the system bottlenecks change over time, for instance, due to the changes in the product-mix, this approach is insufficient.
3. Time profile of workload. If the spatial workload distribution among the different work centers appears to be non-negligible, the temporal workload distribution on each work center, and for the overall system, is no less important. For this reason, the control system must be able to observe a temporal horizon sufficiently large to include several scheduling periods. If the overall workload is utilized by LOMC, the time profile of the overall workload for the system can be easily determined; however, if the work center workload is examined, the evaluation of the workload time

© 2001 by CRC Press LLC

profile for each work center becomes difficult to foresee. This is due to the difficulty in assessing each job arrival time at the different machines when releasing jobs into the system. It should be noted that the time profile of the machine workload must be evaluated to ensure that the system reaches and maintains steady-state operating conditions. The workload time profile for each machine can be exactly determined only by assuming an environment that is deterministic and static in nature. For this reason two different approximate approaches are considered:

- Time-relaxing approach. Any time that a job enters the system, a workload corresponding to the sum of the processing time and setup time for that job is instantaneously, i.e., the actual interval between job release time and the arrival of that job at each machine is relaxed, assigned to each work center included in the process plan for that job.
- Probabilistic approach. Any time that a job is released into the system [Bechte, 1988], a weighted workload corresponding to the sum of the processing time and the set-up time is assigned to each work center included in the process plan for that job. The weight for each machine is determined by the probability that the job actually reaches that machine in the current scheduling period.

4. Load-control methodologies have two different approaches available:

- Machine workload constraints. An upper workload limit for each machine is defined. In this way, an implicit workload balance is obtained among the machines because the job-release policy gives priority to jobs that are to be processed by the underloaded work centers. If a low upper limit for each workload is utilized, low and foreseeable flow-times can be obtained. A suitable trade-off between system flow-time and system throughput should be identified. An increase in the upper workload limit increases the flow-time and, system throughput will increase at a diminishing rate. In general, load limits should be determined as a function of the system performance parameters corresponding to the manufacturing goals determined at the upper planning level. Lower workload limits can be considered in order to avoid machine underloading conditions.
- Workload balancing among machines. This approach allows for job-release into the system only if the workload balance among the machines improves.

Combining the above mentioned parameters in different ways will generate several LOMC approaches. For each approach, performance parameters and robustness must be evaluated through simulation techniques, particularly if the observed technological environment is highly dynamic and probabilistic.

Research Trends in Dynamic Scheduling Simulation Approach

During the 1980s, in the increasingly competitive world of manufacturing, Stochastic discrete event simulation (SDES) was accepted as a very powerful tool for planning, design, and control of complex production systems [Huq et al., 1994; Law, 1991; Pritsker, 1986; Tang Ling et al., 1992a-b]. This enthusiasm sprang from the fact that SDES is a dynamic and stochastic methodology able to simulate any kind of job-shop and FMS. At the same time, SDES is able to respond only to "what if" questions. SDES makes it possible to obtain the output performance variables of an observed system only for a given set of input variables. To search for a sub-optimal solution, it is necessary to check several different configurations of this set of input variables. This kind of search drives the problem into the domain of experimental research methodologies, with the only difference being that the data is obtained by simulation. The expertise required to run simulation studies correctly and accurately exceeds that needed to use the simulation languages, and as a consequence, the data-analysis is often not widely undertaken; therefore, the accuracy and validity of this simulation approach has been questioned by managers, engineers, and planners who are the end users of the information [Montazeri and Van Wassenhove, 1990].

An optimal configuration for a system is generally based on the behavior of several system performance variables. The goal of the simulation study is to determine which of the possible parameters and structural assumptions have the greatest impact on the performance measure and/or the set of model specifications

© 2001 by CRC Press LLC

that leads to optimal performance. Experimental plans must be efficiently designed to avoid a hit-or-miss sequence of testing, in which some number of alternative configurations are unsystematically tried. The experimental methodologies ANOVA and factorial experimental design (FED) can be used to reach this goal [Pritsker, 1986; Law et al., 1991; Kato et al., 1995; Huq et al., 1994; Gupta et al., 1993; Verwater et al., 1995]:

FED is particularly powerful with continuous quantitative parameters [Box and Draper, 1988; Cochran and Cox, 1957; Genova and Spotts, 1988; Wilks and Hunter, 1982]. The main characteristics of FED are:

1. Capability of efficaciously handling a great number of different parameters in limited number of levels. Generally good approximations are possible considering a maximum of three levels for each parameter.
2. Capability to weigh the importance of each independent variable by observing the values and trends for the dependent variables.
3. Capability to determine interaction effects among input variables.
4. Capability to estimate a polynomial function response surface that approximates the unknown correlation among the observed parameters within the observed range.
5. Knowledge, a priori of limitations of the chosen experimental plan. Each experimental plan is related to the degree of the approximating polynomial function. The higher the degree, the better the approximation is and the more complex and expensive the plan is.
6. Minimization of the required number of tests.

ANOVA [Costanzo et al., 1977; Spiegel, 1976; Vian, 1978] observes the behavior of a system by modifying the value of a few parameters m (generally no more than four) that can assume different levels n (particularly effective with qualitative variables). The number of the necessary experiments is equal to the product of number of levels n and the number of parameters m. Measuring the noise present in the experiment, the goal of the analysis is to indicate which of the parameters have more impact on the system, and whether some interaction effects exist between the observed parameters.

Both methodologies, even if very powerful, are very limited in their ability to observe and analyze the input product-mix, which will prove to be a very important system parameter when dynamic scheduling is needed.

Experimental Approach for Simulated Dynamic Scheduling (ESDS)

The proposed ESDS associates appropriate and powerful mathematical-experimental-statistical methodologies with the computer simulation approach to solve a dynamic scheduling problem in a random FMS. ESDS deals with the dynamic and stochastic nature of a random FMS by using a computer simulator to explore different system states. These states are related to specific values of control parameters based on a specific experimental plan. The corresponding system performance variables are correlated with the values of input parameters in order to find an approximate function that links performance variables to input parameters. The analysis of this function will allow the scheduler to extrapolate desired information on the system, such as: sub-optimum solution conditions, stability-region for the FMS versus the change of system parameters (different control policies, different input product-mix, etc.), level of sensitivity of performance variables versus system parameters, interaction among system parameters, and behavior of performance variables versus system parameters.

ESDS mainly utilizes mixture experimental design (MED) and modified mixture experimental design (MMED) instead of factorial experiment design (FED); however, FED is available to ESDS if needed. The differences between these methodologies are outlined below (Cornell, 1990):

- FED observes the effect of varying system parameters on some system response variables. Several levels for the system parameters are chosen. System performance variables are observed when simultaneous combinations of levels are considered. In FED, the main characteristic of the system

© 2001 by CRC Press LLC

parameters is that no mathematical relations or constraints link the different parameters to each other. They are independent and free to vary in their own specific range.

- MED studies the behavior of the system response variables when the relative proportions of the input parameters change in the mixture, without modifying the amount of the mixture itself. In a mixture experiment, the behavior of the system responses is said to depend on the blending property of the input parameters. If the total amount of the mixture is held constant, the behavior of the performance variables is influenced by modifying the values of the relative proportion of the input parameters. In MED, the input parameters are all dependent on each other. They are linked together normally by a common constraint. If x_i represents the level of the i-th input parameter, the following constraint links all the n input variables:

$$\sum_{i=1}^{n} x_i = q \tag{7.11}$$

  where, q is a constant quantity that can be equal to 100, in this case x_i is expressed by percentage values.
- MMED are represented by two types of designs and/or combinations: (a) mixture-amount experimental design is an experimental plan in which the amount of the mixture, as well as the relative proportion of the components can vary. The output variables are assumed to be influenced by both the amount of the mixture and the relative proportion of the components, and (b) mixture-process variables experimental design is an experimental plan in which the process variables can vary as well as the relative proportion of the mixture components. The output variables are assumed to be influenced by both the levels of process parameters and the relative proportion of components.

Mixture experimental designs represent a relatively new branch of statistical research. Almost all statistical literature on this topic has been produced in the last three decades [Sheffe, 1958; Sheffe, 1963; Draper and Lawrence, 1965]. Some scientific fields of application for MED have been: the chemistry industry, the cereal industry, the tire manufacturing industry, the soap industry, the food industry, and any other industry whose product is essentially a mixture. Considering that the use of this methodology has been chemical oriented, the components of the mixture are often called "ingredients".

Chemical mixture and dynamic scheduling problems for a random FMS have a number of similarities. In a dynamic scheduling problem, the variables are not the jobs that must be scheduled, but the following system parameters:

1. Input product-mix (IPM) is represented by the percentage value of each type of product which forms the mix entering the system.
2. Inter-arrival time (IAT) represents the time between two consecutive parts entering the system. The inter-arrival time is a process parameter that directly affects the amount of the total input product-mix. If the value of IAT is held constant, the following formula is given:
   IAT = interval-time/total amount of jobs loaded into the system in the interval-time. When the total amount of jobs loaded into the system in a unit-period coincides with the target system through-put (TST), the following relation exists between the two parameters: IAT = 1/TST. (A stable condition is defined as system output being equal to system input.)
3. Control/dispatching policies. Both queuing and routing rules are qualitative process parameters.
4. Material handling system speed. A quantitative continuos process parameter which primarily influences system performance when there is an absence, or low availability of, storage-queues at the machines.

© 2001 by CRC Press LLC

5. Number of pallets. A discrete quantitative process parameter which constrains the amount of WIP acceptable in the system.
6. Production rate. In many manufacturing activities, the production rates of some machines can be varied by using different tools for the same operation [Drozda and Wick, 1983; Schweitzer and Seidmann, 1989]. The production rate is therefore a discrete quantitative process parameter that can assume different values depending on the tools used.

If, during any time period, the FMS is viewed as a black box process in which a certain amount of mixture of jobs enter the system (whose quantity is determined by the inter-arrival time) and are transformed into finished products at a rate which depends on the above mentioned parameters, then the analogy with a chemical process is complete.

ESDS will use one of the following experimental methodologies, based on the type of the process parameter actually being modified in the study:

FED. If the input-product mix is held constant and some or all of the other parameters are considered variable.

MED. If only the input-product mix is considered variable and all the other parameters are held constant.

Mixture-amount experimental design (MAED). If the input-product mix and the inter-arrival time between two consecutive parts are considered variable and all the other parameters are held constant.

Mixture-process variable experimental design (MPED). If the input-product mix and one or more of the process parameters, i.e., control/dispatching policies, AGV speed, and machine speed, etc., are considered variables, and the remaining parameters are held constant.

Input product-mix, IAT, and dispatching rules are all directly controlled by the PAC system during the scheduling phase. All remaining parameters imply medium-long term decisions (linked with the system layout) that must be undertaken in collaboration between PAC and planning personnel. For this reason, the first three parameters will be the focus in the next sections. It should be made clear that ESDS is capable of handling the complete set of process parameters.

Input Product-Mix and ESDS in a Random FMS

This section will demonstrate that in the case of dynamic scheduling for a random FMS. The actual system throughput related to a specific target throughput is highly dependent on the input product-mix (IPM). An FMS is defined by both its hardware (machines, material handling system, tools, etc.) and software (dispatching rules), while the target through-put corresponds to a specific value of IAT. The potential of the proposed methodology ESDS, which is able to assess an approximate function linking each observed output performance variable with the IPM, will also be demonstrated.

Example Simulated Random FMS

The simulated FMS Figure 7.1 includes five machines (A, B, C, D, and E) and three different Jobs (E1, E2, and E3). Machines A and B for Job E1 and E, and D for Job E2 are interchangeable; therefore, it is possible to identify a total of four alternative paths: two for Job E1 and two for Job E2. This is a type of routing problem. For Job E3, no routing flexibility has been considered. The characteristics of the technological cycle for product E3 can be found in Table 7.1. In an FMS, processing and set-up times of each machine for each job are deterministic due to the automatic nature of an FMS. In this example, a random distribution has been chosen for the above mentioned times because each type of job is representative of a small family of products. These families have small characteristic differences that justify different processing and setup times at the machines. Products are assumed to be uniformly distributed in the family. A random distribution is used to give more generality to the problem. In this way, the ESDS approach can be applied to the more general case of a non-automated job-shop.

© 2001 by CRC Press LLC

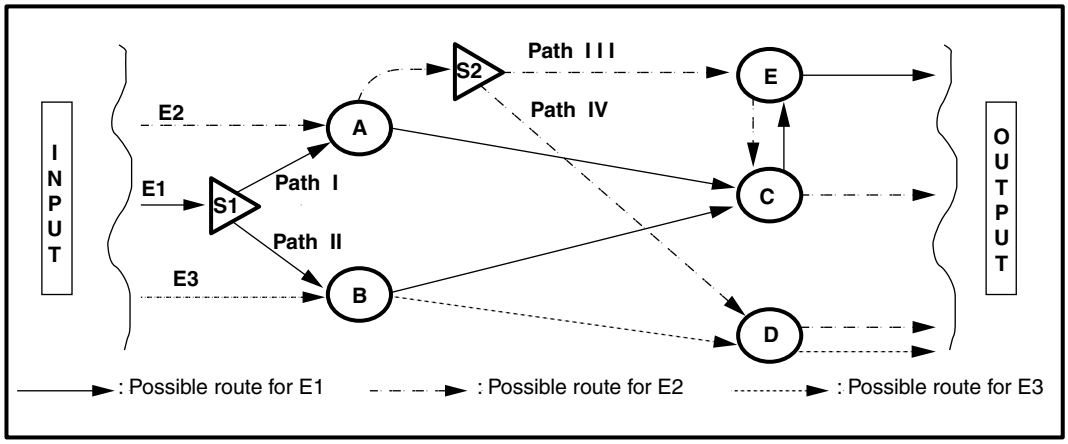


FIGURE 7.1 Random FMS layout.

Mixture Experimental Design (MED) and the Simulated FMS

Some underlining elements of MED are as follows:

1. The number of product types does *not* represent a methodology constraint. If the number of product types is two, ESDS will utilize the traditional FED. It is sufficient to consider the percentage of a product as an independent factor in order to immediately obtain a value for the other products percentage; therefore, in this case, a different combination for IPM will be individualized by the percentage of just one product of the mix. If the number of product types is equal to or greater than three, ESDS will utilize MED or the modified MED designs. However with more than three types of products, the resulting relational function can not be displayed in a 3-D vision diagram.
2. In the general mixture problem (as it occurs in this example) the q component proportions are to satisfy the constraints

$$0 \leq x_i \leq 1$$

$$x_1 + x_2 + \cdots + x_q = 1$$

   All blends among the ingredients are possible.
3. In an actual random FMS, constraints can be applied to variable ranges. In this case, the constraints indicated in 2. will be modified, and constrained-MED methodologies will be employed.

This example deals with three different types of product whose proportions vary from zero to one, with no constraints applying to IPM. The FMS model is also characterized by a constant value for IAT, and fixed values for all other system parameters. Under these assumptions, ESDS employs one of the available MED designs as its experimental plan.

The simplex-centroid design [Sheffe, 1963] is selected over the simplex-lattice design [Sheffe, 1958]. Both designs were introduced in the early 1960s by Sheffe. During this period, the research on mixture experiments was being developed by several researches [Draper and Lawrence, 1965]. Simplex-lattice and simplex-centroid design are credited by many researchers to be the foundation of MED and are still in use today.

A coordinate system used by the experimental designs for mixture is referred to as simplex coordinate system [Cornell, 1990]. With three components, simplex coordinate system is a triangle coordinate system see Figure 7.2 displaying fractional values in parenthesis (x_1, x_2, x_3). The three single component blends are displayed at the vertexes of the triangle where each vertex coincides with the case, and only one type

© 2001 by CRC Press LLC

TABLE 7.1 System Parameters of the Simulated FMS

Arrival Time Between Parts	Norm (5.; 0.5)
Percentage of part E1	Variable
Percentage of part E2	Variable
Percentage of part E3	Variable

Processing Time (min.)

	E1	E2	E3
Machine A	Norm. (15. ; 0.9)	Norm. (3. ; 0.3)	0
Machine B	Norm. (13. ; 0.8)	0	Norm. (17. ; 0.7)
Machine C	Norm. (8. ; 0.6)	Norm. (4. ; 0.3)	0
Machine D	0	Norm. (14. ; 0.9)	Norm. (18. ; 0.7)
Machine E	Norm. (7. ; 0.6)	Norm. (4. ; 0.3)	0

Set-up Time (min.)

	E1	E2	E3
Machine A	Norm. (3. ; 0.3)	2	0
Machine B	Norm. (1. ; 0.3)	0	2
Machine C	Norm. (3. ; 0.3)	Norm (3. ; 0.3)	0
Machine D	0	Norm (8. ; 0.6)	Norm. (6. ; 0.6)
Machine E	Norm. (3. ; 0.3)	2	0

Handling Time (min.)

Path IN -A (E1)	1
Path IN -A (E2)	1
Path IN - B	1
Path A - C	1
Path B - C	1
Path C - E	1
Path A - E	2
Path A - D	1
Path B - D	1
Path E - OUT	2
Path C - OUT	2
Path D - OUT	2

where: Norm (m;v) is equal to Normal Distribution (mean ; variance)

of product enters the FMS. Each side of the triangle represents the case in which a mix of two types of product is loaded into the system; the component represented by the opposite vertex being zero. Every point inside the simplex region represents an input product-mix among the three types of jobs. Three lines, parallel to each side of the triangle and passing through a point under consideration, must be drawn to evaluate the related simplex coordinates, e.g., in Figure 7.5, point A has the following simplex coordinates 0.1, 0.1, 0.8.

Simplex-Lattice Design and Simplex-Centroid Design

An ordered arrangement consisting of a uniformly spaced distribution of experimental points on a simplex is known as a lattice. A lattice may have a special correspondence to a specific polynomial equation. To support a polynomial model of degree m in q components over the simplex, the lattice is referred to as a $\{q, m\}$ simplex-lattice, consisting of several experimental points (the component proportions) whose coordinates are defined by the following algorithm [Cornell, 1990]:

© 2001 by CRC Press LLC

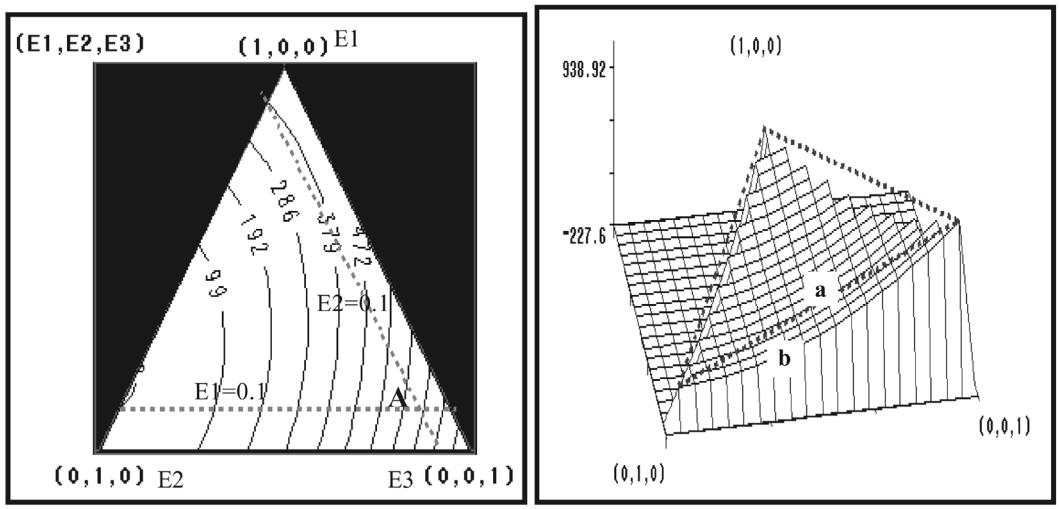


FIGURE 7.2 Contours plot and response surface for the complete predicted model variable FT.

1. The proportions assumed by each component takes $m - 1$ equally spaced values from 0 to 1, that is: $x_i = 0, 1/m, 2/m, \ldots, m/m$.
2. The $\{q, m\}$ simplex-lattice consists of all possible combinations of the previously calculated component proportions.

The number of experimental design points in a $\{q, m\}$ simplex-lattice is:

$$\binom{q+m-1}{m} = (q+m-1)!/m!\,(q-1)! \tag{7.12}$$

For this example, q is equal to 3 (types of products):

- For $m = 2$, the second degree polynomial (see Equation 7.13) will fit the following six points: $\{(x1, x2, x3) = (1,0,0), (0,1,0), (0,0,1), (1/2,1/2,0), (1/2,0,1/2), (0,1/2,1/2)\}$ on the boundary of the simplex triangle:

$$y = \sum_{i=1}^{q} \beta_i x_i + \sum_{i<j}^{q} \sum \beta_{ij} x_i x_j = \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_{12} x_1 x_2 + \beta_{13} x_1 x_3 + \beta_{23} x_2 x_3 \tag{7.13}$$

- For $m = 3$, the third degree polynomial (see Equation 7.14) will fit the following ten points (one being in the center of the triangle): $\{(x1, x2, x3) = (1,0,0), (0,1,0), (0,0,1), (2/3,1/3,0), (2/3,0,1/3), (1/3,2/3,0), (1/3,0,2/3), (0,2/3,1/3), (0,1/3,2/3), (1/3,1/3,1/3)\}$:

$$y = \sum_{i=1}^{q} \beta_i x_i + \sum_{i<j}^{q} \sum \beta_{ij} x_i x_j + \sum_{i<j<k}^{q} \sum \sum \beta_{ijk} x_i x_j x_k + \sum_{i<j}^{q} \sum \delta_{ij} x_i x_j (x_i - x_j) \tag{7.14}$$

In the simplex region, the two expressions Equations 7.13 and 7.14 hold implicitly the quadratic and cubic pure effects in the simple variable x_i. As a matter of fact, for the second-degree polynomial, the

© 2001 by CRC Press LLC

general form is represented by:

$$y = \beta o + \sum_{i=1}^{q} \beta_i x_i + \sum_{i=1}^{q} \beta_{ii} x_i^2 + \sum_{i<j}^{q} \sum \beta_{ij} x_i x_j \qquad (7.15)$$

but if the constraint $\sum_{i=1}^{q} x_i = 1$ is considered, the following identity $x_i^2 = x_i \left(1 - \sum_{\substack{j=1 \\ j \neq i}}^{q} x_j \right)$ can be applied to Equation 7.15, then:

$$y = \beta o * \left(\sum_{i=1}^{q} x_i \right) + \sum_{i=1}^{q} \beta_i x_i + \sum_{i=1}^{q} \beta_{ii} x_i \left(1 - \sum_{\substack{j=1 \\ j \neq i}}^{q} x_j \right) + \sum_{i<j}^{q} \sum \beta_{ij} x_i x_j$$

$$= \sum_{i=1}^{q} (\beta_0 + \beta_i + \beta_i) x_i - \sum_{i=1}^{q} \beta_{ii} x_i \sum_{\substack{j=1 \\ j \neq i}}^{q} x_j + \sum_{i<j}^{q} \sum \beta_{ij} x_i x_j = \sum_{i=1}^{q} \alpha_i x_i + \sum_{i<j}^{q} \sum \alpha_{ij} x_i x_j$$

that is formally equal to Equation 7.13.

In this example, the simplex-centroid design (SCD) has been applied to restrain the number of experiments and still be able to obtain information on the behavior of the system performance inside the simplex region. In a q-component SCD, the number of distinct experimental points is $2^q - 1$. For all of the mixtures, components are given equal proportions. The experimental points are defined by the following algorithm:

- q permutations of (1, 0, 0, 0,…, 0)
- $\binom{q}{2}$ permutations of (1/2, 1/2, 0, 0,…, 0)
- $\binom{q}{3}$ permutations of (1/3, 1/3, 1/3, 0,…, 0)
- up to $\binom{q}{q}$=1 experimental point coinciding with the centroid point (1/q,…, 1/q,…, 1/q)

The related experimental function, Equation 7.16, fitting these experimental points, has the following polynomial expression:

$$y = \sum_{i=1}^{q} \beta_i x_i + \sum_{i<j}^{q} \sum \beta_{ij} x_i x_j + \sum_{i<j<k}^{q} \sum \sum \beta_{ijk} x_i x_j x_k + \ldots + \beta_{12\ldots q} x_1 x_2 \ldots x_q \qquad (7.16)$$

If compared with the previous more complex model Equation 7.15, the polynomial function Equation 7.16 relaxes the cubic coefficient of the binary combinations on the perimeter of the simplex triangular region, but maintain the ternary blending among the components in the interior of the triangle.

In this example, q is equal to 3 (number of product types), and the corresponding experimental points for SCD are:

- 3 permutations of single-component mixture: (1,0,0) (0,1,0) (0,0,1)
- 3 permutations of binary mixture: (1/2,1/2,0) (1/2,0,1/2) (0,1/2,1/2)
- 1 centroid point: (1/3,1/3,1/3)

and the related experimental function that can be inferred to be:

$$y = \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_{12} x_1 x_2 + \beta_{13} x_1 x_3 + \beta_{23} x_2 x_3 + \beta_{123} x_1 x_2 x_3 \qquad (7.17)$$

If the complete quadratic model is able to justify no less than 90% of the experimental variation measured during the experimental plan (R-square $\geq$ 90%), the third-degree coefficient β_{123} is relaxed.

© 2001 by CRC Press LLC

TABLE 7.2 Simulated Mixture-Plan CSD for a Random FMS Under Fixed Dispatching-Rules (FIFO+SNQ) and Variable Input Product-Mix

Input Production-Mix				Performance Variables						Output Production-Mix		
OBS	E1	E2	E3	FT	SSC	SUC	MLQ	ST	COMC	$E1_o$	$E2_o$	$E3_o$
1	0	0	1	1040	62	0.388	246	25.5	0	0	0	1
2	1	0	0	479	27	0.774	37	60	0	1	0	0
3	0	0.5	0.5	261	18.8	0.638	72	70	0.202	0	0.643	0.357
4	0	0	1	1030	62	0.39	244	25	0	0	0	1
5	0	1	0	26	0	0.522	1	86	0	0	1	0
6	0.5	0.5	0	52.8	0	0.89	4	86	0.024	0.517	0.482	0
7	0.5	0	0.5	511	31	0.79	72	57.5	0.079	0.556	0	0.443
8	0	1	0	26	0	0.532	1	87.5	0	0	1	0
9	0.5	0	0.5	480	29	0.788	79	57.5	0.079	0.556	0	0.443
10	0.33	0.33	0.33	312	8.6	0.94	28	76	0.07	0.355	0.368	0.276
11	0.33	0.33	0.33	257	11	0.92	29	75	0.052	0.3	0.373	0.327
12	0	0.5	0.5	248	21	0.618	76	66	0.171	0	0.621	0.379
13	0.5	0.5	0	56	0	0.866	4	86.5	0.045	0.468	0.532	0
14	1	0	0	485	26	0.776	36	60.5	0	1	0	0

Data Analysis

SAS-QC has been used to analyze the data. The following system performance variables were evaluated and the corresponding fourteen observed-values (seven experimental points replicated twice) are reported in Table 7.2:

- Flow-time (FT) has been evaluated as the average of the whole set of flow time values related to each job passing through the FMS during the two observed shifts.
- Steady-state coefficient (SSC) measures the trend of amount of work-in-process in the system. SSC is defined as the difference between the average number of unfinished products in the system during two consecutive shifts (third and fourth shifts in the simulated working-period).
- System utilization coefficient (SUC). The average among the machine utilization coefficients is calculated for each shift. SUC is defined as the average among the whole set of SUCs corresponding to the sequence of simulated shifts.
- Maximum length-queue (MLQ). Defined as the maximum number of jobs to be served waiting in a queue.
- System through-put (ST). Defined as the total number of jobs produced by the FMS during each eight hour shift. ST is calculated without distinguishing between the product types.
- System through-put for each type of job (STJ). Evaluated considering the total number of each type of job produced by the FMS each shift. The related values are not reported in Table 7.2, but they have been used to calculate the proportion of each type of entity produced by the FMS in the two simulated shifts. Corresponding values for each output ratio ($E1_o$, $E2_o$, $E3_o$) are reported in Table 7.2 under the header: Output Product-Mix.
- Controller output-mix coefficient (*COMC*). Defined by the following formula:

$$COMC = \sqrt{\sum_{i=1}^{n} (E_i - E_{io})^2} \tag{7.18}$$

where, E_i is the proportion of the input product-mix for the i-th entity, and E_{io} is the proportion of the output product-mix for the same i-th entity. The higher the *COMC* value, the more mix-instability is present in the system, meaning some product types are naturally preferred by the system over others.

© 2001 by CRC Press LLC

TABLE 7.3 Anova Table for the Predicted Quadratic-Model of the Performance Variable FT

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F	Value
Quadratic Model	6	3384939.03	564156.50	321.42	0.0001	
Error	8	14041.81	1755.23			
Uncorrected Total	14	339898980.84				
R-Square						0.996
FT Mean						375.99

TABLE 7.4 Regression Analysis of the Polynomial Model Parameters of the Performance Variable FT

Parameter	Estimate	T for H_0: Parameter = 0	Pr > |T|	Std Error of Estimate
E1	475.289	16.10	0.0001	29.512
E2	19.289	0.65	0.5317	29.512
E3	1028.289	34.84	0.0001	29.512
E1*E2	-664.172	-4.90	0.0012	135.658
E1*E3	-917.772	-6.77	0.0001	135.658
E2*E3	-969.772	-7.15	0.0001	135.658

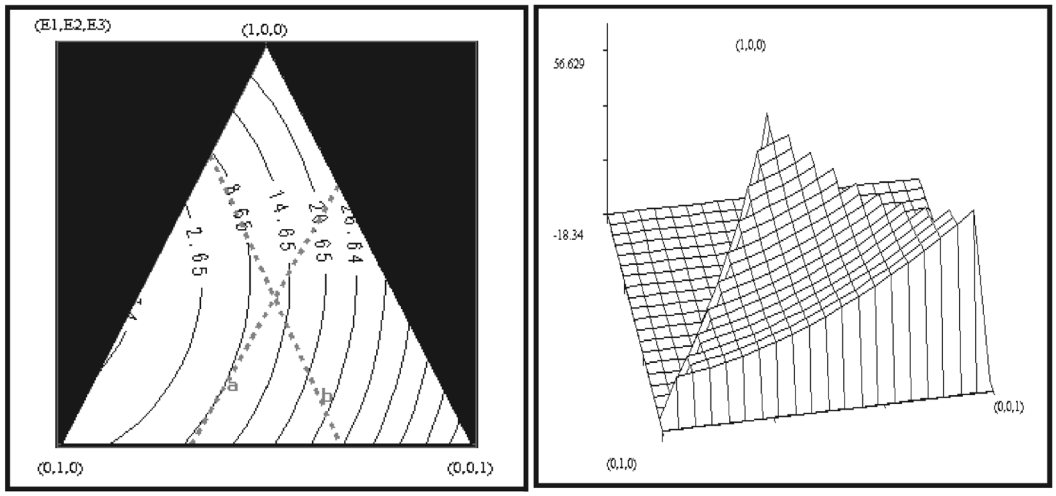


FIGURE 7.3 Contour plots and response surface for the complete predicted model variable SSC.

Flow-Time

ANOVA (see Table 7.3) for the flow-time (FT) reveals that the quadratic model, which justify 99.6% of the total experimental variation, fits the data exceedingly well. The regression analysis for the quadratic model (see Table 7.4) shows that all coefficients, except the coefficient related to variable E2, are important with P-values < 0.002. The main effect term related to the quantity of Job E2 in the system can be relaxed in the model because the proportion of this element in the input product-mix does not directly influence FT. At the same time, Job E2 strongly influences the flow-time variable when the interaction effects with the other two jobs are considered. An increase in the proportion of this job in the input product-mix has a positive effect on FT (both interaction coefficients E2E3 and E1E2 are negative). This is displayed in Figure 7.3, in which both diagrams show a decrease in FT as the combination moves closer to the single component blend E2 (0,1,0). Figure 7.2 also demonstrates how an increase of Job E3 in the input product-mix is poorly tolerated by the system, resulting in a rapid increase of FT. The simulated random

© 2001 by CRC Press LLC

TABLE 7.5 Results Obtained from Minimax Ridge Search Applied on the Complete Predicted Model for SSC

Simplified Model for the Performance Variable SSC, after the Regression Analysis				
SSC = 26.673 E_1 + 62.173 E_3 - 56.469 E_1E_2 - 60.469 E_1E_3 - 47.869 E_2E_3				
where E_1, E_2, E_3 are input product-mix proportions of the entities in the system				
Minimum Area evaluated with a ridge search on the complete predicted model for the performance variable SSC				
E1	E2	E3	PRED	STD. ERR.
0.41951	0.51381	0.06667	−0.0791	0.78462
0.43239	0.56287	0.00474	−2.0696	0.90783
Maximum Point evaluated with a ridge search on the complete predicted model for the performance variable SSC				
E1	E2	E3	PRED	STD. ERR.
0.07912	0.02450	0.89638	52.3960	0.7633

FMS, (under the stated conditions of, control policies, inter-arrival time, etc.) shows a good attitude in transforming Jobs E2 and E1, and difficulties in transforming Job E3. A minimax ridge search confirms this trend. Minimum flow-time (best result) is obtained for the input product-mix (36.3%, 63.2%, 0.5%) and a maximum flow-time (worst result) is obtained for the input product-mix (10.6%, 0.4%, 89.0%).

System-Stability and Mix-Instability

Previously, the problem related to system stability and the relationship between the system stability and dynamic scheduling-control policies have already been highlighted. In this section, the goal is to underline the fact that system stability is a function of the input product-mix as well. SSC has been employed to observe the trend of WIP over time. If WIP increases for an input product-mix, then the value of SSC is different from zero and the system has not reached a steady-state at the end of the last considered shift (the fourth shift in the simulated FMS). The corresponding input product-mix can not be used for a long period, and the consistency of the other system output performance variables is questionable, since the related predicted values also change with time. Contour plots and response surface for the associate second degree polynomial function are displayed in Figure 7.3. For the example random FMS, a high level of instability is associated with a high proportion of Job E3 in the input product-mix. Similar considerations can be driven from the minimax ridge search (see Table 7.5).

The input product-mix can be different from the output product-mix in proportion when dynamic scheduling control is applied; this is defined as system mix instability. Some products can remain in the system longer than others. As a result, the output product-mix changes the proportion of its components over time. In this example, a Controller Output-Mix Coefficient defined as $\text{COMC} = \sqrt{\sum_{i=1}^{n}(E_i - E_{io})^2}$, tests for the presence of mix-instability. COMC ranges between 0 to 1, zero when the input product-mix is equal to the output product-mix. Acceptable values change from system to system. In the example FMS, a COMC value close to 0.07 is considered acceptable. Such a value can correspond to differences in proportion approximately equal to 5% between input and output product-mixes for two types of jobs, with the third job having the same proportional makeup. The quadratic model is considered sufficient to explain the total variation present in the experimental data even if the adjusted R-square is smaller than the R-square values obtained for the other performance variables. Regression analysis shows that only two interaction effects are significant for the model to fit the experimental design results: E1E3 and E2E3. Figure 7.4 displays how the mix instability varies as the proportion of the input product-mix changes.

This example system is sensitive to variables *E*2 and *E*3. Input product-mix is equal to output product-mix for the combinations close to the minimum point (48%, 47%, 5%), whereas mix instability is observed for combinations close to the maximum point (4%, 45%, 54%) (see Table 7.6). The example that has been presented shows that system stability and mix instability of any random FMS controlled with dynamic scheduling policies depends on the proportions of the input product-mix.

© 2001 by CRC Press LLC

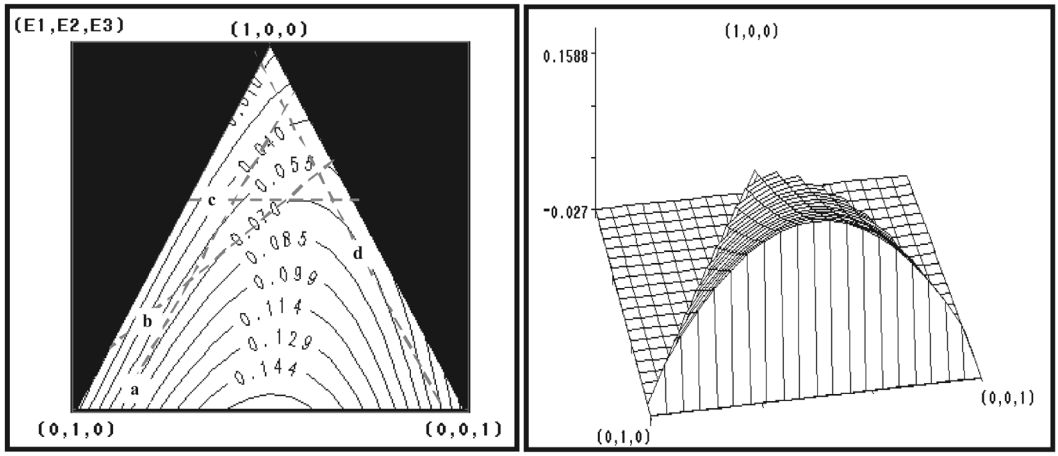


FIGURE 7.4 Contours plot and response surface for the complete predicted model variable COMC.

TABLE 7.6 Results Obtained from Minimax Ridge Search Applied on the Complete Predicted Model for COMC

Simplified Model for the Performance Variable COMC, after Regression Analysis
$COMC = 0.2174\ E_1E_3 + 0.6474\ E_2E_3$
where E_1, E_2, E_3 are input product-mix proportions of the entities in the system
Minimum Point evaluated with a ridge search on the complete predicted model for the performance variable COMC

E_1	E_2	E_3	PRED	STD. ERR.
0.47893	0.47045	0.05062	0.03449	0.017631

Maximum Point evaluated with a ridge search on the complete predicted model for the performance variable COMC

E1	E2	E3	PRED	STD. ERR.
0.00436	0.45282	0.54282	0.16465	0.019665

Physical Constraints on the System

Maximum-length queue (MLQ) is an output performance variable that reflect how queue-lengths vary with different input product-mixes (see Figure 7.5 and Table 7.7). The main interaction effect of E2 can be neglected because the related coefficient is not significant. Job E3 is poorly tolerated by this example FMS, as indicated by the fact that the related coefficient is positive and the highest. The SUC represents the workload distribution on the shop-floor and can be used to verify the level of utilization of the FMS, which will help satisfy economical considerations (see Figures 7.6 and 7.7).

System Through-Put (ST)

To calculate the system through-put for each type of job for a specific input production-mix, ST is multiplied by the proportional ratio of the considered type of job in the output product-mix (see Table 7.2). Figure 7.8 demonstrates that the system through-put, related to a specific target through-put in a random FMS, and controlled by dynamic scheduling, is a function of the input product-mix.

ST varies from a predicted minimum of 35 parts/shift to a maximum of 88 parts/shift (see Table 7.8). Observed values range from 25 parts/shift to 87.5 parts/shift (see Table 7.2). These values can be questionable in the areas of the simplex, where the FMS is not stable, and can be intrinsically misleading for any kind of scheduling methodology to assume a priori the value of the system through-put to be

© 2001 by CRC Press LLC

TABLE 7.7 Results Obtained from Minimax Ridge Search Applied to the Complete Predicted

Simplified model for the performance variable MLQ, after regression analysis

$MLQ = 37.0682\ E_1 + 245.5682\ E_3 - 70.3636\ E_1E_2 - 272.3636\ E_1E_3 - 207.3636\ E_2E_3$

where E_1, E_2, E_3 are input product-mix proportions of the entities in the system

Minimum Area evaluated with a ridge search on the complete predicted model for the performance variable MLQ

E_1	E_2	E_3	PRED	STD. ERR.
0.42674	0.56523	0.00803	0	2.48835

Maximum Point evaluated with a ridge search on the complete predicted model of the performance variable MLQ

E_1	E_2	E_3	PRED	STD. ERR.
0.00514	0.02545	0.96941	231.802	2.51423

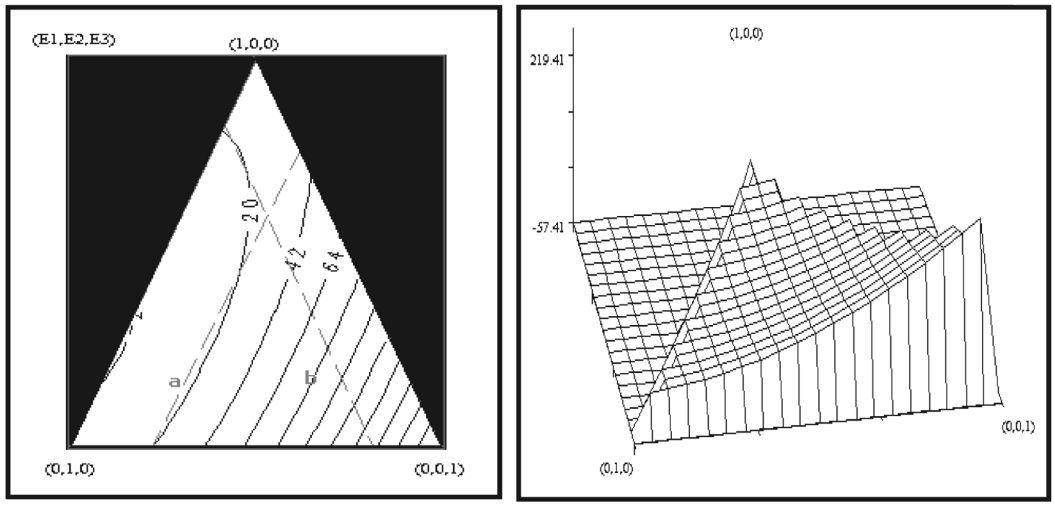


FIGURE 7.5 Contours plot and response surface for the complete predicted model variable MLQ.

known. Unless system through-put is known through historical data referring exactly to the same FMS under the same conditions, i.e., dispatching policies, inter-arrival time, input product-mix, etc., then the utilization of that value to schedule jobs into the system and forecasts system behavior will produce inaccurate results.

Mix-Oriented Manufacturing Control (MOMC)

After analyzing all of the diagrams and general considerations derived from the previous sections, the problem is now how all this information can be effectively utilized to solve the dynamic scheduling problem. It has been shown that all of the main system performance variables (system through-put, system stability, mix-instability, WIP, queue-lengths, system utilization coefficient, etc.) are significantly correlated with the input product-mix (IPM). The knowledge, even if approximated, generated by ESDS of the analytical relationship between each output performance variable and the IPM, gives us the ability to control and modify the system parameter in IPM. This ability, in turn, allows us to consider a new dimension in the dynamic scheduling of a random FMS called MOMC.

© 2001 by CRC Press LLC

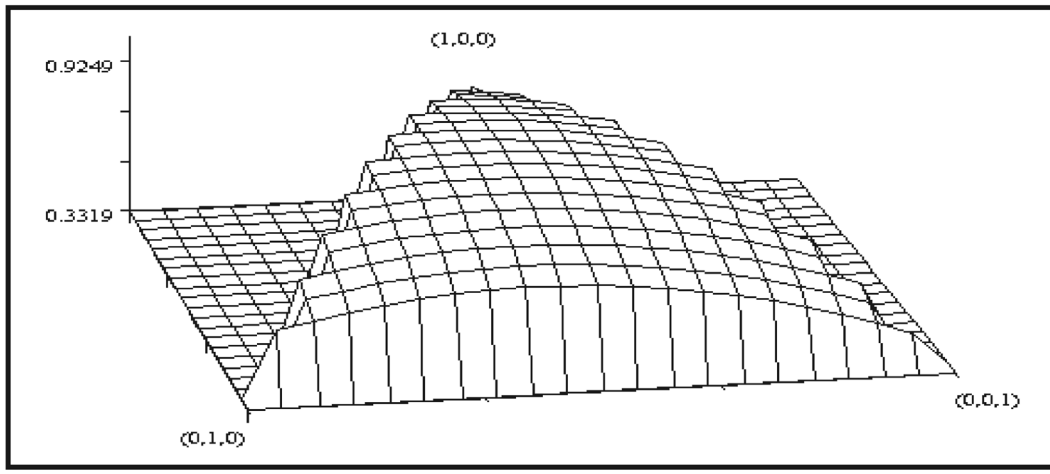


FIGURE 7.6 Response surface for the complete predicted model variable SUC.

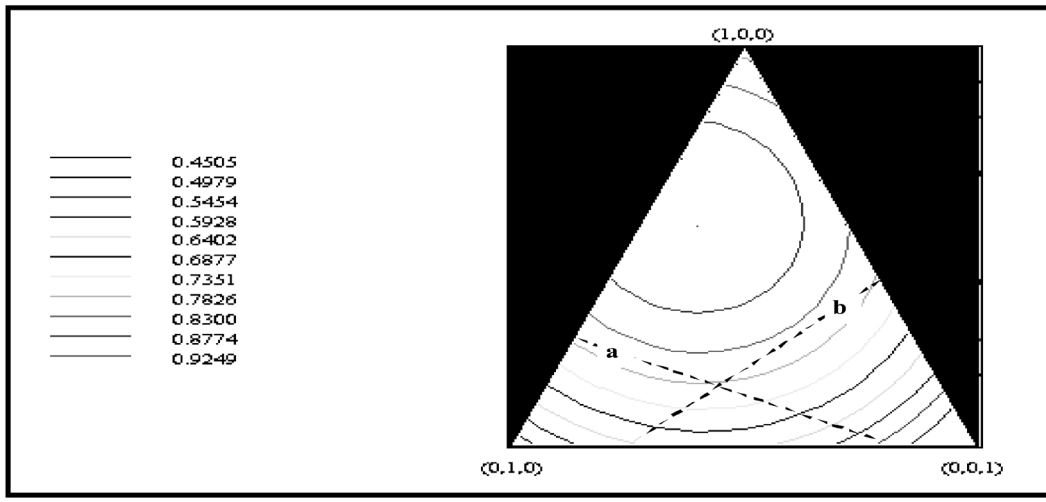


FIGURE 7.7 Contours plot for the complete predicted model variable SUC.

In every manufacturing system, the following different types of product-mixes can be considered:

1. Medium-term output mix (MTOM). Output product-mix required to be produced in medium term (weeks/month).
2. Instantaneous input product-mix (IPM). Instantaneous input product-mix of job-requests entering the system in short-term (hours/shift).
3. Instantaneous output mix (IOM). Actual output product-mix produced in short-term (hours/shifts).

MTOM is defined at the superior planing level by the master requirements plan (MRP). IPM can be defined at the scheduling level by the production active control (PAC). IOM is the actual outcome of the planning and scheduling activities. IPM can be utilized by PAC as a system parameter to dynamically control and schedule a FMS effectively. To generate a robust schedule, continuous feed-back between the planning stage and the scheduling stage is necessary. If assumptions made at upper level (planning stage) are not acceptable at lower level (scheduling stage), then the generated MTOM cannot be produced by

© 2001 by CRC Press LLC

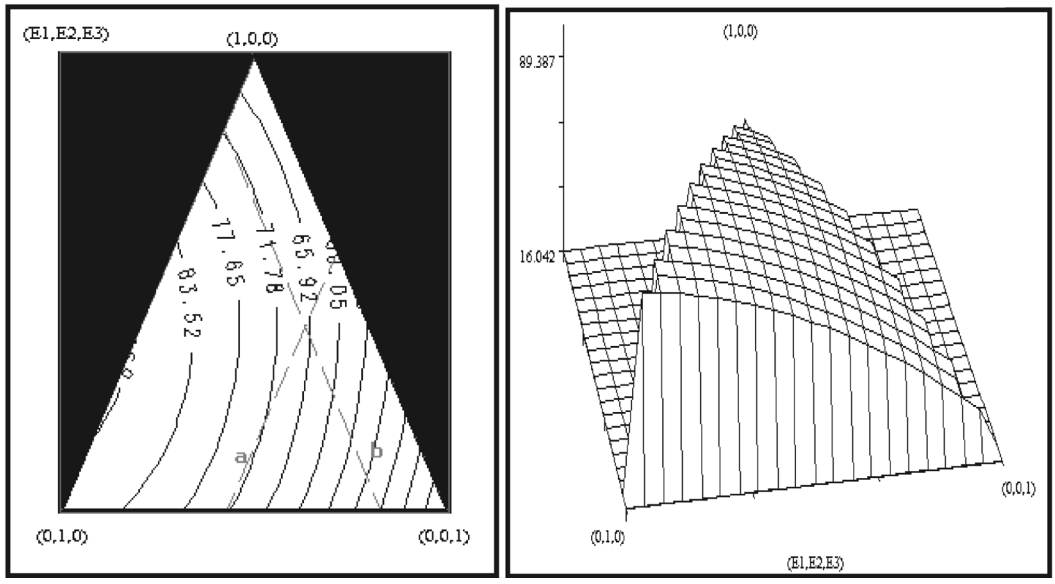


FIGURE 7.8 Contours plot and response surface for the complete predicted model variable ST.

TABLE 7.8 Results Obtained from Minimax Ridge Search Applied to the Complete Predicted Model for ST

Simplified model for the performance variable ST				
$ST = 60.21\ E_1 + 86.71\ E_2 + 25.21\ E_3 + 51.72\ E_1E_2 + 59.72\ E_1E_3 + 48.72\ E_2E_3$				
where E_1, E_2, E_3 are input product-mix proportions of the entities in the system				
Minimum Point evaluated with a ridge search on the complete predicted model for the performance variable ST				
E1	E2	E3	PRED	STD. ERR.
0.08168	0.02241	0.89591	34.8945	0.61983
Maximum Point evaluated with a ridge search on the complete predicted model for the performance variable ST				
E1	E2	E3	PRED	STD. ERR.
0.42357	0.56648	0.00995	87.8136	0.72401

the FMS, and bottlenecks and/or instability problems become unavoidable. MOMC handles this cyclical link and provides robust/quasi-optimal schedules.

MOMC for Batch-Production

During the generation of the MRP document, customer orders are decomposed into the required different parts. Each part is assigned to the related industrial department to be produced. In this way, each department has its own set of parts that constitute its MTOM. The resulting MTOM is characterized by: total production quantity and proportion mix. These two attributes are deeply correlated to each other; the first one being the result of the latter.

In our example, MTOM can be represented by the following matrix MTOM $= (q_1,d_1; q_2,d_2; q_3,d_3)$, where q_i is the quantity of product type i required for due-date d_i (see Table 7.9). Two different heuristic algorithms are presented in the following sections. In these algorithms, the product-mix and the inter-arrival time between jobs are the only modifiable system parameters, meaning that the other system parameters (control policies, processing times at the machines, AGV speed, etc.) are considered not to be modifiable.

Backward Heuristic Algorithm (BHA)

BHA is similar to the approach commonly used to develop a MRP document. However, the previous use of ESDS, with its related results, allows for the definition of an algorithm that is capable of providing

© 2001 by CRC Press LLC

TABLE 7.9 MTOM Matrix for an Hypothetical Order Request Starting a Time t*=to Model for MLQ

MTOM	Quantity q_i (units)	Due Date d_i (weeks)	Initial Stock S_{io}	Final Stock S_{if}
E1	1035	3	0	0
E2	1840	4	0	0
E3	300	2	0	0
Gant Chart for the required products				
	I week	II week	III week	IV week
E1				
E2				
E3				

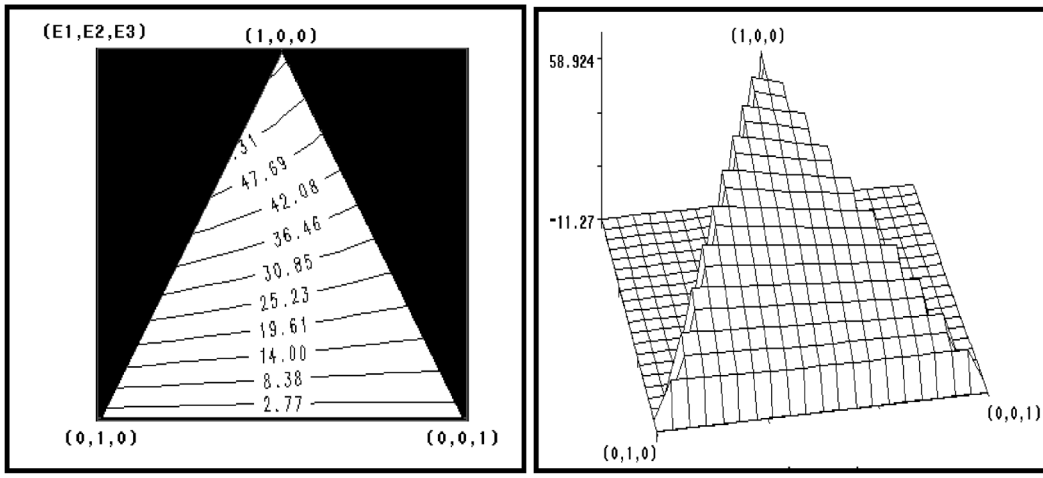


FIGURE 7.9 Contours plot and response surface for the predicted model variable STE_1.

more robust non-optimum solutions. Given the matrix MTOM (see Table 7.9) composed of two column vectors $\boldsymbol{Q}(q_1, q_2, q_3)$ and $\boldsymbol{D}(d_1, d_2, d_3)$, in which each element of $\boldsymbol{Q}$ represents a quantity, and each element of $\boldsymbol{D}$ represents a due-date, assume a known initial total target through-put (TT). The inter-arrival time between jobs is evaluated using TT and ESDS, as applied to the simulated FMS. In this way, the output variables STE_i, WIP, SSC, SUC, and MLQ are evaluated over the simplex region for the considered TT (see Figures 7.9, 7.10, and 7.11 for STE_i). Our example FMS has the following physical constraints related to SSC (1) and MLQ (2), which identifies an acceptable area ABCD marked in Figure 7.12:

1. $SSC(t) = f(E_{in}) = 26.67\ E_1 + 62.17\ E_3 - 56.47\ E_1E_2 - 60.47\ E_1E_3 - 47.87\ E_2E_3 \leq$ low constant = (e.g. 8.66)
2. $MLQ(t) = f(E_{in}) = 37.07\ E_1 + 245.57\ E_3 - 70.36\ E_1E_2 - 272.36\ E_1E_3 - 207.36\ E_2E_3 \leq$ constant = (e.g. 20)

Side CD of area ABCD has been defined by constrain Equation 7.3, which refers to the SUC:

3. Constant = (e.g. 0.735) $\leq SUC(t) = f(E_{in}) = 0.77\ E_1 + 0.52\ E_2 + 0.38\ E_3 + 1.04\ E_1E_2 + 0.96\ E_1E_3 + 0.81\ E_2E_3 \leq 1$

This constraint can generally be relaxed in a real world problem, unless the SUC is so low that it is not economically convenient to utilize the available FMS to produce the required product-mix. From the physical system constraint diagrammed (WIP, SSC, SUC, and MLQ) the acceptable area ABCD (see Figure 7.12) can be identified in the simplex region. In the area *ABCD*, an average approximate value for

© 2001 by CRC Press LLC

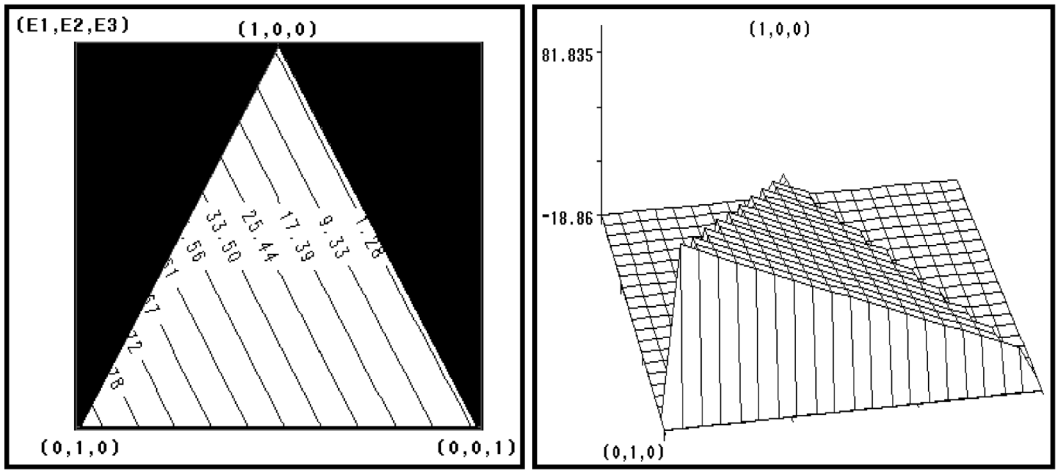


FIGURE 7.10 Contours plot and response surface for the predicted model variable STE_2.

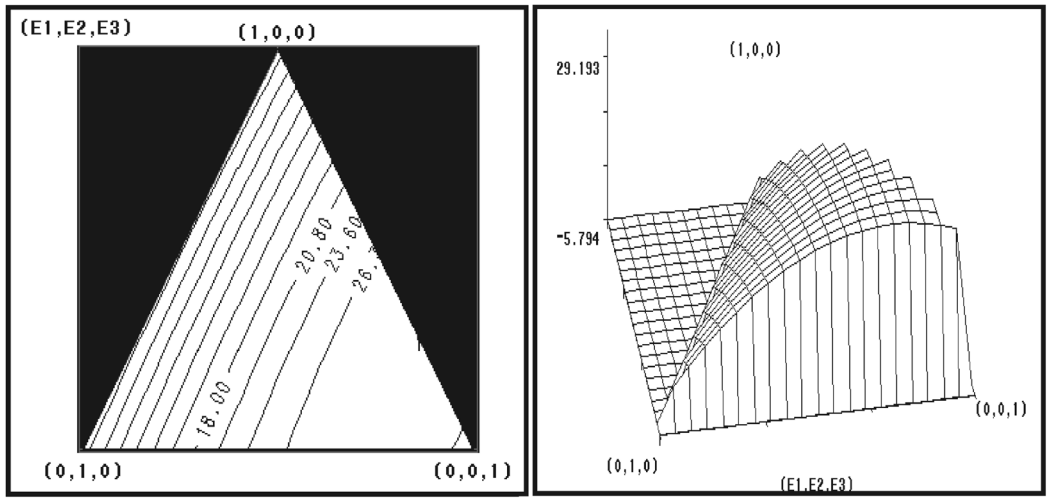


FIGURE 7.11 Contours plot and response surface for the predicted model variable STE_3.

each system through-put STE_i can be assessed using the performance diagram (STE_i) (see Figure 7.13) and Formula 7.19:

$$\overline{STEi} = \frac{\iint_{ABCD} STEi \; ds}{\iint_{ABCD} ds} \tag{7.19}$$

Using the STE_i as determined by Formula 7.19, the lead-time (LT_i) for each type of product batch can be calculated as: $LT_i = q_i / STE_i$. The latest-start time (LST_i) for product type i-th can be calculated by using each due-date di in the following formula: $LST_i = d_i - LT_i$. A resulting Gantt chart, similar to that one represented in Table 7.9, shows the calculated LT and LST value combinations for the planned operations.

© 2001 by CRC Press LLC

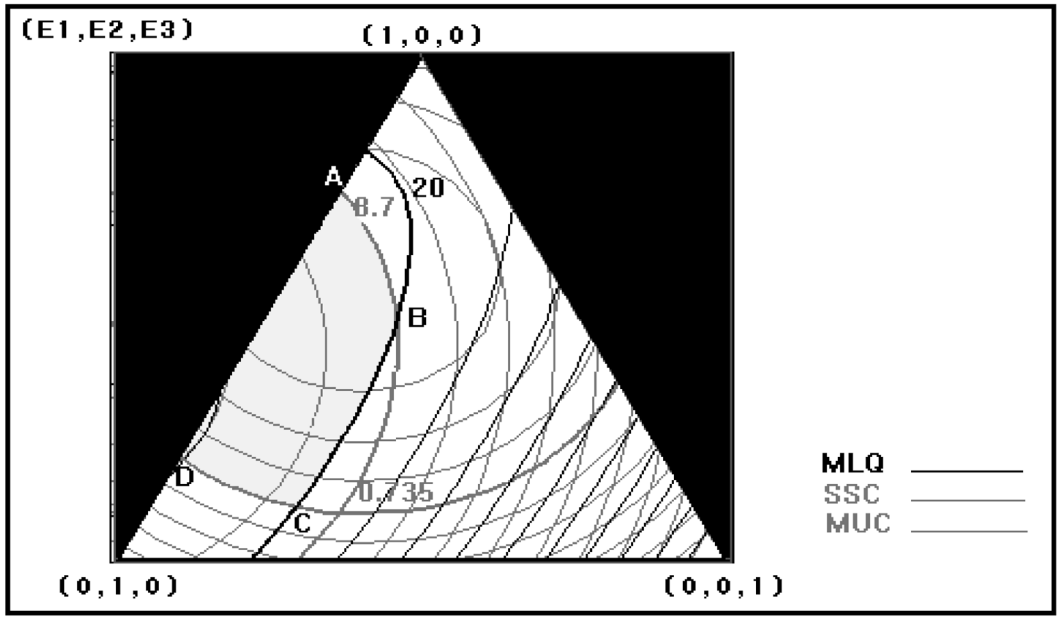


FIGURE 7.12 Acceptable area ABCD which verifies the considered constraints.

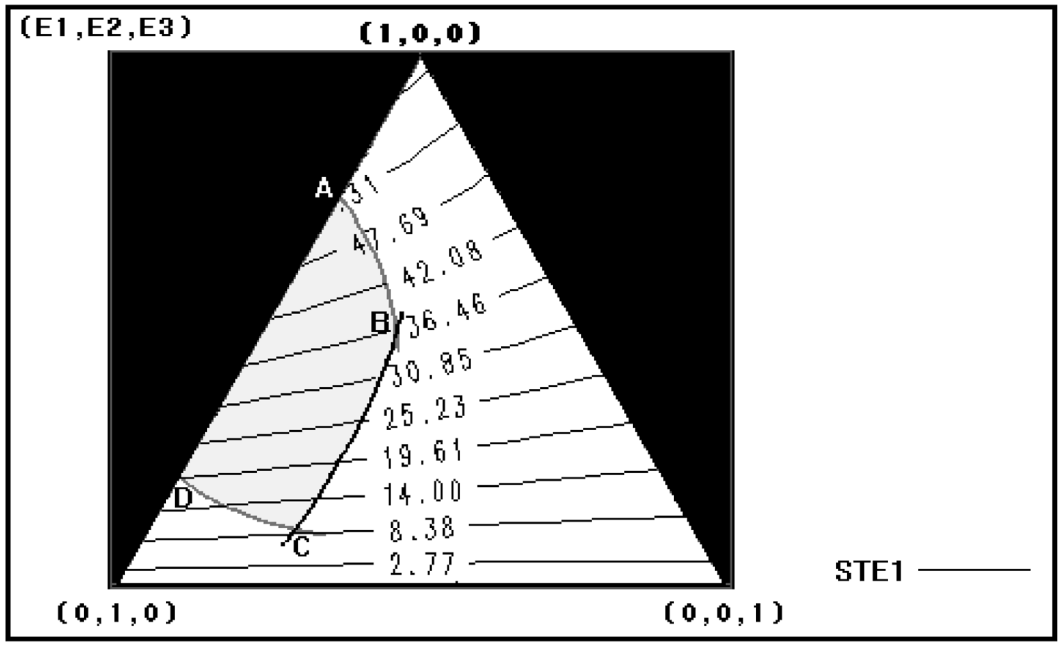


FIGURE 7.13 Example of STE_i trend in the acceptable area ABCD.

The following situations are possible:

- One of the evaluated LST_i falls before "time 0" planning initial time. In which case, the BHA algorithm must be repeated with an increased value for *TT*. If *TT* can not be increased, then the MTOM cannot be implemented by the FMS in the required time interval. Other alternatives, e.g., sub-commission, choose of product to be delayed in delivery, etc., must be considered.

© 2001 by CRC Press LLC

- All the evaluated LST_i fall after the "time 0". For each interval time ΔT all the necessary system performance variables are now known, so the actual q_i produced in ΔT and the related product-mix $P(x_1, x_2, x_3)$ can be calculated. The following situations are now possible:

P is inside the *ABCD* area. In this case, all the calculations performed are acceptable, the product-mix P is feasible, and the schedule can be sent to the production system.

P is not inside the *ABCD* area. First, adjust LST_i for some operation (moving the calculated LST_i backward towards the origin of the axis) in such a way that the resulting product-mix is acceptable. If this fails, start with the interval time in which the mix is not acceptable. For this interval, increase the *TT* and perform a new cycle of calculation to re-evaluate system behavior. If no *TT* can be increased in any ΔT, then the MTOM cannot be implemented by the FMS in the required time interval. Other alternatives, (e.g., sub-commission, choice of product to be delayed in delivery, etc., must be considered.

Forward Heuristic Approach (FHA)

Just as in BHA, FHA has a matrix MTOM composed of two column vectors $\mathbf{Q}(q_1, q_2, q_3)$ and $\mathbf{D}(d_1, d_2, d_3)$, and an initial total TT (see Table 7.9). The inter-arrival time between jobs is evaluated using TT. ESDS is applied to the simulated FMS. The output variables STE_i, WIP, SSC, SUC, and MLQ (see Figures 7.9, 7.10, and 7.11) are evaluated over the entire simplex region for a given TT. Starting from "time 0", the initial planning time, the STE_i function for the product with the earliest due-date is evaluated and set equal to the through-put required to satisfy the product demand (q_i). For instance, in the example used in this chapter: $STE_3 = f_a(x_1, x_2, x_3) = q_3/T_3 = 300/20 = 15$ (units/shift). In this case, time T_3 is measured in working-shifts (WS), such that WS = (d_i)* (number of working days) *(number of shifts per work day). The intersection of the response surface STE_3, and the plan parallel to the base of the simplex region in which $z = 15$ (units/shift), has the property of being the set of product-mix combinations which is equal to the required demand. However, regarding Curve C, only the set of points inside area ABCD are of interest for the evaluation of system performance. This set of points is called the definition set (DS) (see Figure 7.14). DS can be moved rigidly into the other simplex regions referring to the system performance variables STE_2 and STE_1 This geometrical procedure has a numerical correspondence procedure correlated to it.

Following a managerial strategy for this example, precedence has been given to the product having the next due date, e.g., E_1. The point belonging to the set DS which maximizes the system through-put for the product E_1 is determined to be point B (see Figure 15). Three different values of STE_i corresponds to point B ($STE_1 = 39$, $STE_2 = 25.5$, $STE_3 = 15.2$). These values are used to determine the total amount of products $P = (780, 510, 304)$ that can be produced during intervals T_I and T_{II}, while not violating system constraints or due dates for the earliest required product, e.g., E_1. The remaining quantity R, to be produced in the next interval T_{III}, is equal to $Q - P = (255, 1330, 0)$. The FHA algorithm can be repeated under this new condition for the time interval T_{III}. Two situations are equally possible:

1. The remaining quantity r_i of the next product to be produced (in this example Job E_1) cannot be produced in the required inter-arrival time (T_{III}). In this case, TT must be increased and the results recalculated. If TT can not be increased, the FMS is not able to produce the required total amount of products under the considered constraints. Managerial options must be reconsidered and evaluated.
2. The remaining quantity r_i of the next due date product (E_1) can be produced in the required time interval (T_{III}). In this case, the next step will be to proceed to the next time interval T_{IV}. If during any time interval T_i the required amount of product j-th cannot be produced, then one of two solutions can be attempted:

Increase the STE_j related to the j-th product in an interval time $T_i - 1$ in such that the remaining amount of production r_j to be produced during interval time T_i is decreased. Note that an increase of STE_j (in $T_i - 1$) results in a decrease of the other STE_i (per $i \neq j$). The satisfaction of the other product-constraints must be verified and maintained.

© 2001 by CRC Press LLC

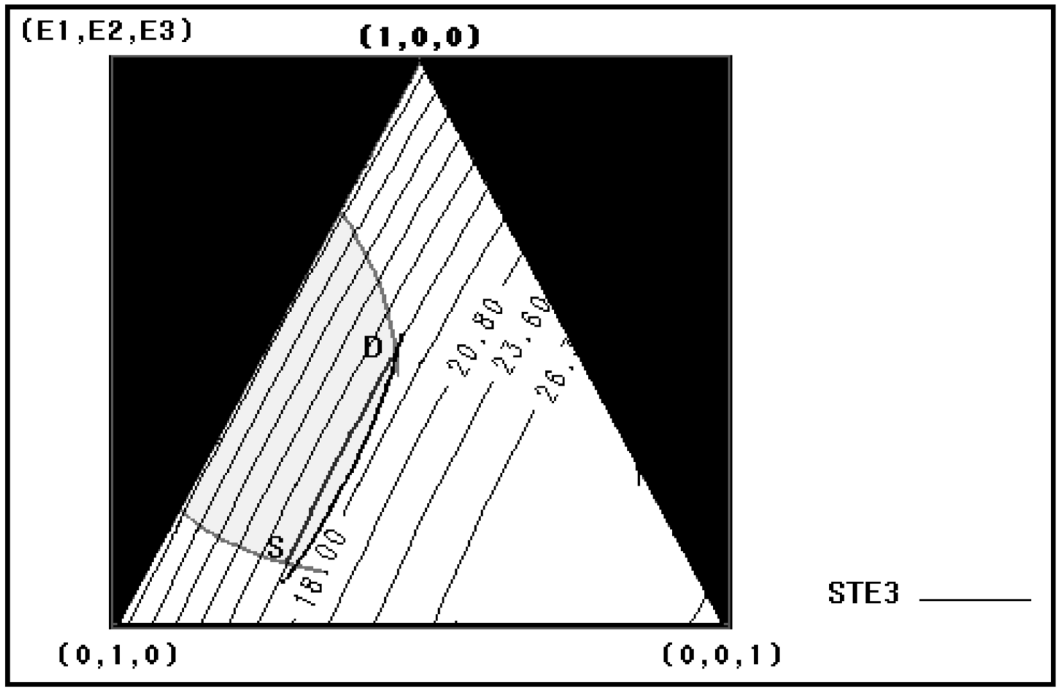


FIGURE 7.14 Definition set (DS) for the variable STE_3.

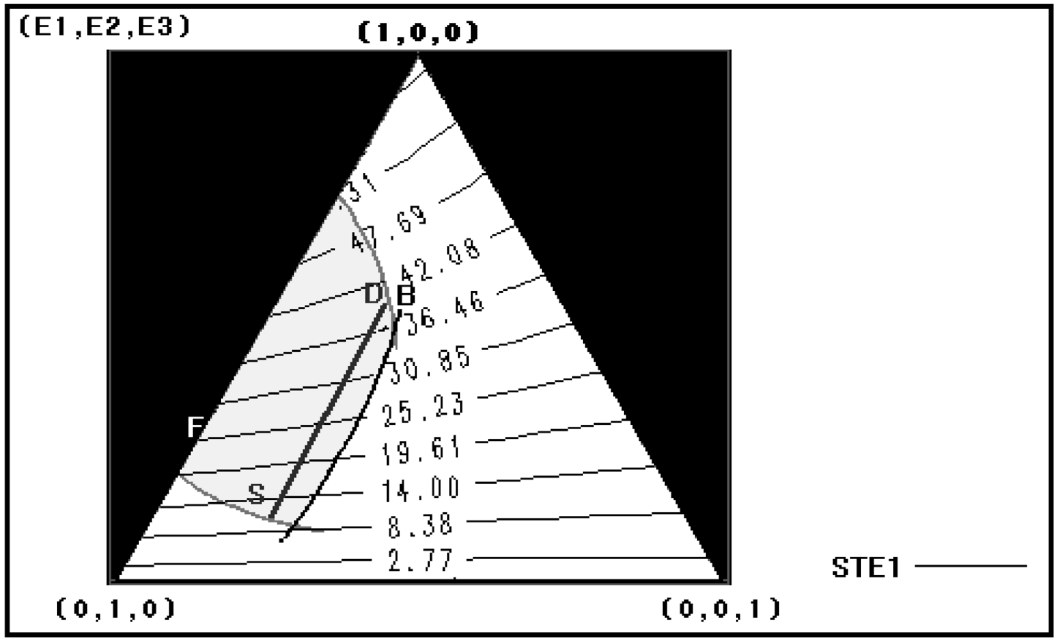


FIGURE 7.15 Identification of point B for the variable STE_1.

© 2001 by CRC Press LLC

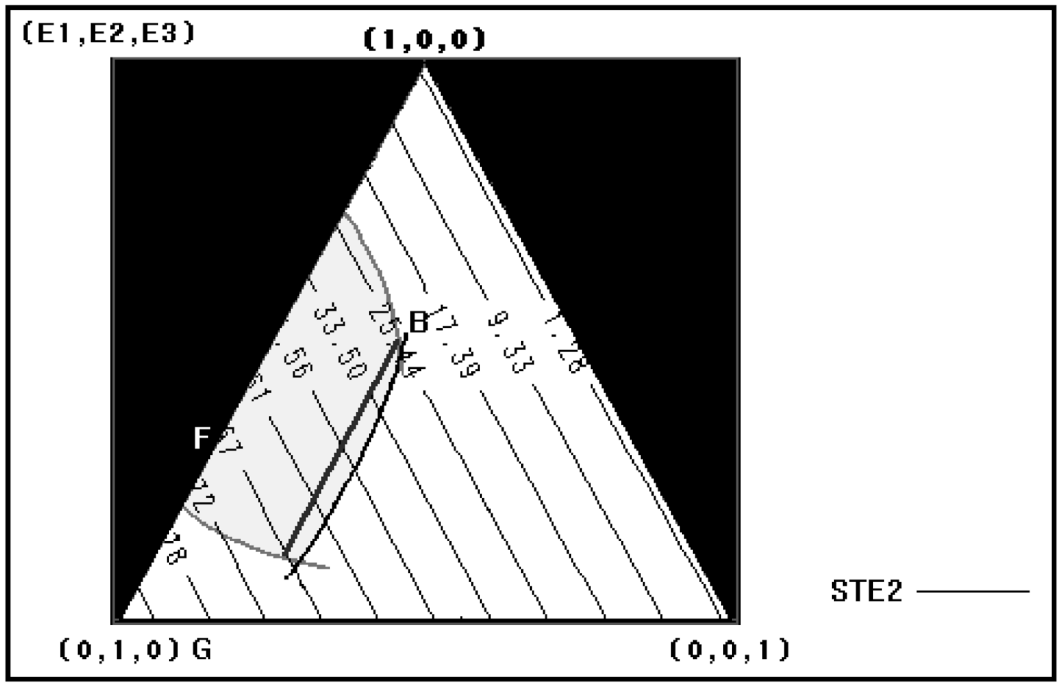


FIGURE 7.16 Point B for variable STE_2.

Increase the TT in the current and/or previous T_i and repeat the calculations.

If neither solutions are possible, the problem is not solvable. The required amount of product-mix can not be processed in the example FMS and others managerial solutions need to be considered. Once this algorithm has scheduled all of the products, the required product-mix is acceptable. At this point the resulting Gantt chart, with the amount of parts and the starting dates for each parts-batch, can be given to the PAC system to start the actual production.

In this example, at the end of the interval time T_{III}, product E_1 must be delivered: $STE_1 = (r_i / T_{III}) = (255/10) = 25.5$. Point F in Figure. 7.15 represents the required solution: F belongs to the contour $STE_1 = 25.5$ and to the side of the simplex region (coinciding with the case in which only the two products E_1 and E_2 are produced by the FMS). In Figure 7.15, the variable STE_2 assumes the value 57.6 units/shift corresponding to the point F. Vector Q in interval time T_{III} is equal to: $Q(255, 576, 0)$ and vector $R(0, 754, 0)$. In time interval IV, the remaining vector R gives a required system through-put for job E_2 equal to: $STE_2 = (754/10) = 75.4$. The required through-put is less than the actual throughput obtainable by the system in case of unique production of Job E_2 this situation corresponds to point G in Figure 7.16, where $STE_2 = 83.4$. The fact that point G falls out of the region ABCD in Figure 7.16 is not important because the constraint violated is SUC. This constraint is an inferior limit which can be relaxed in this model; therefore, the system can produce the remaining quantity r_2 in time interval T_{IV}. The completed schedule is acceptable and can be sent to the PAC system for the actual start of operations.

The two heuristic approaches provide non-optimum solutions, but they are robust and maintain both economical and physical constraints. Furthermore, the two approaches follow two opposite directions (backward and forward) to solve the same planning problem, leading to solutions that can be different but acceptable. This gives the manager the ability to choose a third solution that is between the two extremes proposed by MOMC. Finally, by implementing these algorithms in an expert system, the computational burden of the whole methodology can be alleviated.

© 2001 by CRC Press LLC

MOMC Conclusions

The use of MOMC at the scheduling phase is necessary for two reasons:

1. Validation of the proposed schedule determined during the planning phase. This control is strongly suggested, even if the results are often positive. At the scheduling phase, the real status of the system at the beginning of operations is known; whereas, during the planning phase, the status can only be assumed or hypothesised. Knowledge of the initial status of a system allows a scheduler to more accurately simulate a system in order to verify the results obtained during the planning phase. If the product-mix and the related job-schedule is still acceptable, the PAC system will start production. Otherwise, when necessary, modifications will be made to the IPM in order to achieve the target matrix MTOM.
2. Unforeseen changes to system status during the planning phase due to the dynamic nature of the random FMS environment. Examples of these changes include: raw materials delays, machine breakdowns, a lack of tools to process an operation, changes in due-dates, changes in required amount for i-th product, strikes, maintenance, etc. In all of these cases, the generation of a new schedule is necessary. For some cases, e.g., materials delays, due dates, and production changes, only external constraints must be modified while the system configuration does not change, i.e., the simulated system is still valid. Also, in these cases the system performance functions are still equal to the original ones. In other cases, e.g., machine breakdowns, maintenance, lack of tools, etc., some internal system characteristics change with the result that a new system must be simulated. New system performance functions must be evaluated and utilized to provide the new schedule.

7.3 Summary

In this chapter we have looked at a large number of issues relating to the problem of scheduling in the flexible manufacturing environment. First, performance parameters that are appropriate for an FMS were covered. Next we looked at the background of manufacturing systems, along with their characteristics and limitations. FMS technology and the issue of flexibility were also discussed in some detail. We covered both static and dynamic scheduling, in addition to a number of methods for dealing with these scheduling problems. The third major section of this chapter outlined a new approach to these issues called MOMC. Research trends, an experimental approach for ESDS along with operational details and examples were also covered in this section.

References

Baker, K. R., 1974, Introduction to sequencing and scheduling, John Wiley & Sons, New York.

Balas, E., 1965, An additive algorithm for solving linear programs with zero-one variables, *Operations Research,* n.13, pp. 517–546.

Balas, E., 1967, Discrete programming by the filter method, *Operations Research,* n.15, pp. 915–957.

Balas, E., 1970, Machine sequencing, disjunctive graphs and degree-constrained subgraph, *Naval Research Logistics Quarterly,* n.17, pp. 941–957.

Barash, M. M. et al., 1976, The optimal planning of computerized manufacturing systems, *NSF Report,* April 74–25256.

Barret, R. T. and Barman, S., 1986, A SLAM II simulation of a simplified flow shop, *Simulation,* n.47, pp. 181–189.

Bechte, W., 1988, Theory and practice of load-oriented manufacturing control, *Int. J. Prod. Res.,* vol. 26, n.3, pp. 375–395.

Box, G. E. P. and Draper, N. R., 1988, Empirical model-building and response surfaces, John Wiley & Sons, New York.

© 2001 by CRC Press LLC

Caroyer, C. and Liu, Z., 1991, Effectiveness of heuristics and simulated annealing for the scheduling of concurrent tasks — an empirical comparison, Institute Nationale de Recherche en Informatique, Le Chesnay, France.

Cochran, W. G. and Cox, G. M., 1957, Experimental Designs, John Wiley & Sons, New York.

Cornell, J. A., 1990, *Experiments with Mixtures,* John Wiley & Sons, New York.

Costanzo, A. et al., 1977, *Statistics Lesson,* Elia, Rome.

Davis, L., 1991, *Handbook of Genetic Algorithms*, Van Nostrand Reinhold, New York.

Della, Croce F., Tadei, R. and Volta, G., 1992, A genetic algorithm for the job shop problem, *D.A.I. Politecnico di Torino,* Italy.

Dorigo, M., 1989, Genetic algorithms: the state of the art and some research proposals, Dipartimento di Electronica, Politecnico di Milan, Italy.

Draper, N. R. and Lawrence, W. E., 1965, Mixture design for three factors. *J. Royal Statistical Soc.,* B, v. 27, pp. 450–465.

Drozda, T. J. and Wick, C., 1983, *Tool and Manufacturing Engineers Handbook,* v. 1, Society of Manufacturing Engineers, Dearborn, MI.

Falkenaur, E. and Bouffoix, S., 1991, A genetic algorithm for job shop, *Proceedings of the 1991 IEEE International Conference on Robotics and Automation.*

Florian, M., Trepant, P., and McMahon, G., 1971, An implicit enumeration algorithm for the machine sequencing problem, *Management Science,* n.17, pp. 782–792.

Fox, M. S. and Smith, S. F., 1984, ISIS-A knowledge-based system for factory scheduling, *Expert System,* n.1, pp. 25–49.

Fry, T. D., Cox, J. F., and Blackstone, J. H., 1992, An analysis and discussion of the optimized production technology software and its use, *Production and Operation Management,* n. 1, pp. 229–242.

Garey, M. R. and Johnson, D. S., 1979, Computers and intractability: an introduction to the theory of NP-completeness, W. H. Freeman & Co., San Francisco.

Genova, J. R. and Spotts, F., 1988, Statistics for manufacturing engineers, The University of Chicago Press, Chicago.

Glaser, R. and Hottenstein, M., 1982, Simulation study of a closed-loop job shop, *J. Operations Management,* n.3, pp. 155–166.

Glover, F. and Laguna, M., 1989, Target analysis to improve a tabu search method for machine scheduling, in *Advanced Knowledge Research Group,* US West Advanced Technologies, Boulder, CO.

Glover, F., 1990, Tabu search: a tutorial, *Interfaces,* n.20 (4), pp. 74–94.

Gomory, R. E., 1965, On the relation between integer and non-integer solutions to linear programs, *Proceedings of the National Academy of Science,* n.53, pp. 160–265.

Gomory, R. E., 1967, Faces of an integer polyhedron, *Proceedings of the National Academy of Science,* n.57, pp. 16–18.

Graves, S. C., 1981, A review of production scheduling, *Operations Research,* v.29 n.4, pp. 646–657.

Greenberg, H., 1968, A branch-and-bound solution to the general scheduling problem, *Operations Research,* n.16, pp. 353–361.

Gupta et al., 1993, Operations planning and scheduling problems in advanced manufacturing systems, *Int. J. Prod. Res.,* v. 31, n. 4, pp. 869–900.

Hadavi, K. et al., 1990, ReDS-a dynamic planning, scheduling, and control system for manufacturing, *J. Manuf. Syst.,* vol. 9, no. 4, pp. 332–344.

Holland, J. H., 1975, *Adaptation in Natural and Artificial Systems,* University of Michigan Press, Ann Arbor.

Huq et al., 1994, Process oriented manufacturing system simulation to measure the effect of shop-control factors, *Simulation,* v. 62, n. 4, pp. 218–228.

Ignall, E. and Schrage, L. E., 1965, Application of the branch-bound technique to some flow shop scheduling problems, *Operations Research,* n.13, pp. 400–412.

Ishibuchi, H., Tamura, R., and Tanaka, H., 1991, Flow Shop Scheduling by Simulated Annealing, *Transactions of the Institute of Systems, Control and Information Engineers,* n.4, pp. 111–117.

© 2001 by CRC Press LLC

Kato et al., 1995, Integrated approach for loading, routing and scheduling in FMS, *IEEE Symposium on Emerging Technologies & Factory Automation,* v. 1, IEEE, pp. 299–310, Piscataway, NJ.

Kirkpatrick, S., Gelatt, C.D., and Vecchi, M. P., 1983, Optimization by simulated annealing, *Science,* 220, pp. 671–680.

Kumar, A. and Morton, T. E., 1991, Dynamic job routing in a heavily loaded shop, *GSIA,* Carnegie Mellon University, Pittsburgh.

Laguna, M., Barnes, J. W., and Glover, F., 1989, Scheduling jobs with linear delay penalties and sequence dependent setup cost using tabu search, Department of Mechanical Engineering, University of Texas, Austin.

Law, A. M. et al., 1991, *Simulation Modeling & Analysis*, 2nd ed., McGraw-Hill, New York.

Liu, J. and MacCarthy, B. L., 1996, The classification of FMS scheduling problems, *Int. J. Prod. Res.,* vol. 34, n.3, pp. 647–656.

Lowerre, B. T., 1976, The harpy speech recognition system, Ph.D. thesis, Carnegie Mellon University, Pittsburgh.

Lundrigan, R., 1986, What is the thing they call OPT?, *Production and Inventory Management,* n. 27 (second quarter), pp. 2–12.

Mattfeld, D.C., 1996, Evolutionary search and the job shop: investigations on genetic algorithms for production scheduling, Physica-Verlag, Heidelberg.

McDuffie, E. L. et al., 1995, Using scheduling in flexible manufacturing systems, *Engineering Applications of Artificial Intelligence: The International Journal of Intelligent Real-Time Automation,* vol. 8, n. 6, pp. 681–688 (December).

McDuffie, E. L. et. al., 1996, Scheduling and temporal dependent interval calculus, *Intelligent Scheduling of Robots and Flexible Manufacturing Systems,* pp. 183–190, Institute of Technology, Arts and Science, The Center for Technological Education Holon, CTEH Press.

McMahon, G. B. and Burton, P. G., 1967, Floor-shop scheduling with the branch-and-bound method, *Operations Research,* n. 15, pp. 473–481.

Meleton, M. P., 1986, OPT-fantasy or breakthrough?, *Production and Inventory Management,* n. 27, (second quarter), pp. 13–21.

Melnyk, S. A. and Carter, P. L., 1987, *Production Activity Control*, Dow Jones-Irwin, Homewood, Ill.

Montazeri, M. and Van Wassenhove, 1990, Analysis of scheduling rules of an FMS, *Int. J. Prod. Res.,* vol. 28, no. 4, pp. 785–802.

Morton, T. E. and Pentico, D. W., 1993, Heuristic scheduling system, John Wiley & Sons, New York.

Morton, T. E., Lawrence, S. R., Rajagopolan, S., and Kekre, S., 1988, SCHED-SAR: a price-based shop scheduling module, *J. Manuf. Operations Manage.,* n.1, pp. 131–181.

Nakano, R. and Yamada, T., 1991, Conventional genetic algorithm for job shop problems, *Proceedings of the 4th International Conference on Genetic Algorithms,* pp. 474–479, San Diego.

Ow, P. S. and Morton, T. E., 1988, Filtered beam search in scheduling, *Int. J. Prod. Res.,* n.26, pp. 297–307.

Pai, A. R. and McRoberts, K. L., 1971, Simulation research in interchangeable part manufacturing, *Management Sci.,* n.17, pp. 732–743.

Pritsker, A. A. B., 1986, *Introduction to Simulation and SLAM II*, Systems Publishing Corporation, West Lafayette, IN.

Rachamadugu, R. and Stecke, K. E., 1994, Classification and review of FMS scheduling procedures, *Production Planning & Control,* vol. 5, no. 1, pp. 2–20.

Rubin, S., 1978, The argos image understanding system, Ph.D. thesis, Carnegie Mellon University, Pittsburgh.

Schweitzer, P. J. and Seidmann, A., 1989, Processing rate optimization for FMSs with distinct multiple job visits to work centers, in *Proceedings of the Third ORSA/TIMS Conference of Flexible Manufacturing Systems,* K. E. Stecke and R. Suri, Eds., Elsevier Science, Amsterdam.

Schwimer, J., 1972, On the n-job, one-machine, sequence-independent scheduling problem with tardiness penalties: a branch-bound solution, *Management Science,* n.18, pp. 301–313.

© 2001 by CRC Press LLC

Sheffe' H., 1963, The simplex-centroid design for experiment with mixtures, *J. Royal Stat. Soc.,* B, v. 23, pp. 235–263.

Sheffe' H., 1958, Experiment with mixtures, *J. Royal Stat. Soc.,* B, v. 20, pp. 344–360.

Smith, S. F., 1987, A constrain-based framework for reactive management of factory schedules, in *Intelligent Manufacturing: Proceedings of the International Conference on Expert Systems and Leading Edge in Production Planning and Control,* M.D. Oliff, Ed., Benjamin-Cummings Publishing, Menlo Park, CA.

Spiegel, M. R., 1976, *Statistics,* SCHAUM, Etas, Milan, Italy.

Syswerda, G. and Palmucci, J., 1991, The application of genetic algorithms to resource scheduling, Fourth International Conference on Genetic Algorithms.

Tang Ling, Lang et al., 1992a, A framework for part type selection and scheduling model in FMS environments, research memorandum, n. 92–12, (June) School of Industrial Engineering, Prude University.

Tang Ling, Lang et al., 1992b, A study on decision rules of scheduling model in an FMS, research memorandum, n. 92–17, (July) School of Industrial Engineering, Prude University.

Van Laarhoven, P. J. M., Aarts, E. H. L., and Lenstra, J. K., 1992, Job shop scheduling by simulated annealing, *Operations Research,* n.40, pp. 113–125.

Vepsalainen, A. and Morton, T. E., 1988, Improving local priority rules with global lead-time estimates, *J. Manuf. Operations Manage.,* n.1, pp. 102–118.

Verwater et al., 1995, Computer-aided development of flexible batch production recipes, *Production Planning and Control,* v. 6, n. 4, pp. 320–330 (Jul.-Aug).

Vian, F., 1978, *Statistics: Analysis of Variance and Experimental Plan*, CLEUP, Padova.

Vollman, T. E., 1986, OPT as an enhancement to MRP II, *Production and Inventory Management,* n. 27 (second quarter), pp. 38–47.

Vollman, T. E., Berry, W. L., and Whybark, D. C., 1992, *Manufacturing Planning and Control Systems,* IRWIN, New York.

Widmer, M. and Herts, A., 1989, Tabu search techniques: a tutorial and an application to neural networks, *OR Spectrum,* n.11, pp. 131–141.

Wilks, S. S. and Hunter, J. S., 1982, Introductory engineering statistics, John Wiley & Sons, New York.

Wolfe, W. J., Sorensen, S. E., and McSwain, J., 1997, Planning and Scheduling Algorithms, University of Colorado Technical Report, January 26.

Wolfe, W. J., 1995a, Spacebased resource planning — phase II, *Summary Report for Hughes Information Technology Systems Research Grant #SG-258799SJP,* (Jan.) Aurora, Colorado.

Wolfe, W. J., 1994, Spacebased resource planning-phase I, *Summary Report for Hughes Information Technology Systems Research Grant #SG-258799SJP,* (Jan.) Aurora, Colorado.

Wolfe, W. J., 1995b, Scheduling with distribution criteria, *Report for Hughes Information Systems Corp., Research Grant #SG-258799SJP,* (Nov.) Aurora, Colorado.

© 2001 by CRC Press LLC

8

Automated Systems Techniques for On-Line Quality and Production Control in the Manufacture of Textiles

8.1 Introduction

8.2 Automation of Basic Textile Processes
Automation of Spinning • Automated Systems in Weaving • Automated Systems in Finishing

8.3 Distributed Systems for On-Line Quality and Production Control in Textiles
Basic Concepts for On-line Control in Textiles • Approaches to Building Cost Effective Real-Time Control Systems in Textiles • Software Realization • Integrating Control and Manufacturing Systems in Textiles

8.4 Summary

Stantcho N. Djiev
Technical University of Sofia

Luben I. Pavlov
Technical University of Sofia

8.1 Introduction

Textile manufacturing involves a variety of sequential and parallel processes of continuous and discrete nature. Each requires precise, on-line control of preset technological parameters such as speed, pressure, temperature, humidity, and irregularity. On the manufacturing sites, these processes take place within separate machines or production lines where a relatively large number of operating personnel and workers are engaged. The intensities of the material flows: raw materials (fibers, yarn, and sliver), dyes, chemicals, ready production, etc., are substantially high, and this leads to heavy transport operations, inevitably involving costly hand labor.

The raw materials processed in textile possess poor physical and mechanical properties concerning tensile strength, homogeneity, and others. This causes frequent stops in the technological process due to thread breaks, engorgement, winding of the material around rollers, etc. As a result, labor-consuming and monotonous hand services are required for the proper operation of each textile machine. Statistics show that due to higher productivity and new technologies, the total number of machines at an average textile factory has decreased more than twice in recent decades [Baumgarter et al., 1989]. Nevertheless, the problem for replacing hand labor in textile manufacturing still remains a challenge in all aspects of process automation.

© 2001 by CRC Press LLC

Textile materials usually undergo many technological passages, leading to greater energy consumption and large amounts of expensive wastes, some of which may be recycled within the same process.

Taking into account the above mentioned characteristics of the textile production as a whole, the following approaches for application of automated systems techniques in the field can be outlined.

- Creation of new processing technologies and development of new generations of highly automated textile machines.
- Application of highly efficient controlling and regulating microprocessor-based devices, integrated in to distributed control systems. This would ensure reliability of information and allow the implementation of standard industrial fault-tolerant information services.
- Usage of industrial robots and manipulators for automation of the basic and supplementary operations, resulting in increased productivity and lower production costs.
- Automation of transport operations for reducing the amount of hand work and process stops which often occur when sequential processes are badly synchronized. The optimization of machine speeds and loading is an important source for higher efficiency throughout textile manufacturing.
- Development and implementation of new concepts and informational and control strategies, so that the highly automated and computerized CAD, CAM, CAP, and CAQ sub-systems can be totally integrated, forming a Computer Integrated Manufacturing (CIM) or a Computer Aided Industry (CAI) system. The resulting systems are not just a mixture of sub-structures, but process internal informational homogeneity, common software tools, databases, and other features. Usually, these systems are developed using systematic approach techniques. The CIM and CAI super systems and the level of their internal integration should be considered on the basis of the specific, and often contradictory, requirements of textile manufacturing.

The development of automated systems in textiles, as a whole, can be summarized in the following four stages:

The first stage is characterized by partial automation of separate machines or operations using conventional controlling devices. Such examples are the pick finders and cop changers in the weaving machines, local controllers of temperature, speed, pressure, etc. At this stage, a large percentage of handwork is still used.

The second stage involves usage of automated systems for direct (most often digital) control of the technological process. This stage requires a greater reliability level of the equipment due to the centralized structure and remote mode of operation and data processing of these systems. Hand-labor is reduced by means of manipulators, robots, and automated machines. The automated control subsystems collect information from various objects and pass it to a central control unit while retaining control over the following.

- Continuous control of local process parameters.
- Timing registration and basic statistics for machines stops, idle periods, malfunctioning, etc.
- The local systems produce alarm signals, or even stop machines for the operators if their abnormal operation affects the quality beyond preset limits or when dangerous situations occur.
- Some indirect qualitative and quantitative indices are calculated or derived: materials and energy consumption, quality parameters of the ready production, actual or expected (extrapolated) amounts of wastes, etc.

As a result, the central control unit produces and sends information in the form of data sheets, protocols, and recommendations to the operating personnel. This information is also stored and retrieved later for off-line decision support when optimizing and planning the material flows, machines loading, etc.

The third stage is characterized by implementation of direct numerical control of many or all technological variables using dedicated and totally distributed control systems. The term distributed here does not represent only the spatial dispersion of the control equipment, but rather, the fully autonomous

© 2001 by CRC Press LLC

mode of action of each controlling/measuring node while it is still connected with other devices through the industrial network.

Local control units for data acquisition, processing, and retrieval, combined with intelligent field sensors, substantially increase the reliability of the automated system as a whole. The latter is usually built on a hierarchical principle, incorporating within itself several independently working layers. Nearly fully automated production lines are implemented at this stage using high production volume machines running at variable speeds, so that a total synchronization is achieved throughout. Computerized subsystems like CAD, CAQ, CAP, and others are implemented at this stage to different extents. There exists here some integration among them, using local area networks (LANs) and wide area networks (WANs). As a whole, the production facilities, although highly automated, do not yet exhibit substantial integration.

The fourth stage involves the integration of the production in computer-integrated manufacturing (CIM) or computer-aided industry (CAI) systems. Due to the specific features of textiles and the dynamic changes in the stock and labor markets, this stage still remains a challenge for future development and will be discussed later in this chapter.

8.2 Automation of Basic Textile Processes

Automation of Spinning

Bale-Opening and Feeding Lines

In the preparatory departments of the textile mills take place actions for bale-opening and feeding of the card machines. The transportation and unpacking of the incoming bales, e.g., cotton bales and ready laps involved much hand labor in the recent past. As an alternative, an automated cotton bale-opening machine is shown in Figure 8.1. It comprises two main assemblies: a motionless channel (see 10 in figure) for the cotton transportation and a moving unit (4) for taking off the material. This unit is mounted on the frame, (13)

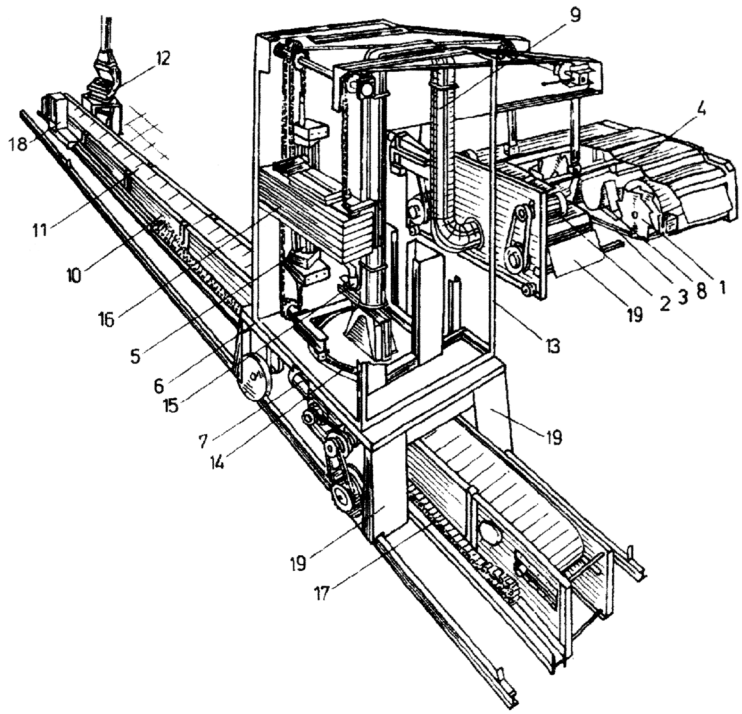


FIGURE 8.1 Automated bale-opening machine.

© 2001 by CRC Press LLC

which slides down the railway alongside the transportation channel. The cotton bales are placed on both sides of the channel. Approximately 200 bales with different sorts of cotton of variable height can be processed simultaneously. The take-off unit (4) is programmed in accordance with the type of the selected mixture. It takes off parts of the material by means of the discs (1), actuated by the AC motor (2). The depth of penetration into the bale is controlled by the rods (3). The pressing force of the unit (4) is controlled according to the readings of a pneumatic sensor. The signal is forwarded to a microprocessor controller (usually a general-purpose PLC) which commands a pneumatic cylinder (5) to change the elevation of the unit (4). The material then goes into pneumo-channels (8,9) and the transportation channel (10). The subsequent machines are fed through the channel (12) by means of a transporting ventilator. A magnetic catcher placed inside the channel (12) prevents the penetration of metallic bodies into the feeding system. The take-off unit (4) moves along the railway at a speed of 0.1–2.0 m/s, driven by the AC motor (7). It can make turns of 180 degrees at the end of the railway and then process the bales on the opposite side. The frame (13) and the bearing (14) accomplish this, while the position is fixed by the lever (15). The productivity of these machines approximates 2000 kg/h, and they usually feed up to two production lines simultaneously, each of them processing a different kind of textile material mixture.

Automation of Cards

Figure 8.2 shows a block diagram of an automated system for control of the linear density of the outgoing sliver from a textile card machine. The linear density [g/km] is measured in the packing funnel (2). The sensor signal is processed in the controller module (3/14), which governs the variable-speed drive (4) by changing the speed of the feeding roller (5); thus, long waves of irregularity (over 30 meters in length) are controlled. The regulator also operates the variable-speed system (6), which drives the output drafting coupled rollers (7) of the single-zone drafter (8). Long-term variances of the sliver linear density are suppressed by the first control loop. The winding mechanism (9) rotates at constant speed and provides preset productivity of the card.

Automation of Drawing Frames

The growing intensification of contemporary textile production resulted in the development of high-speed drawing frames for processing the textile slivers after the cards. The output speeds of the drawing frames often reach 8–15 m/s. This, together with the high demands for product quality, brings to life new techniques for development, and implementation of automatic control systems for on-line quality and production

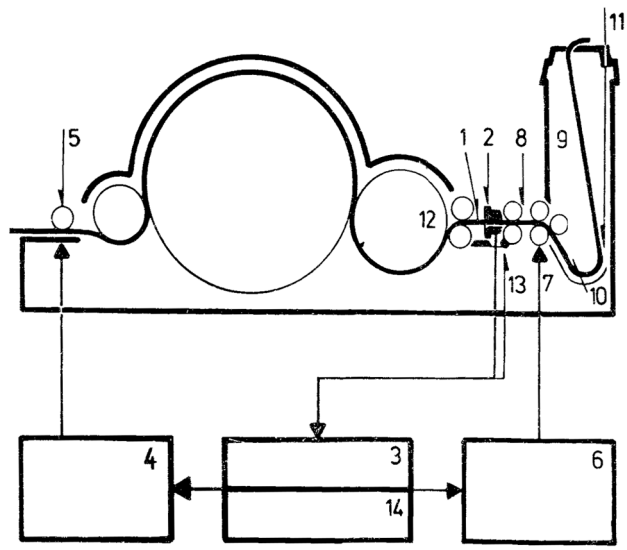


FIGURE 8.2 Card with automatic control of the output sliver linear density (closed-loop control system).

© 2001 by CRC Press LLC

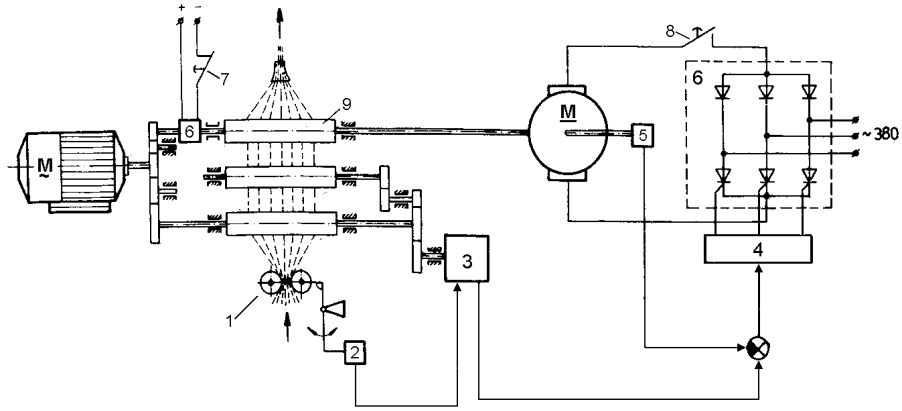


FIGURE 8.3 Drawing frame with open-loop automatic control of the sliver linear density.

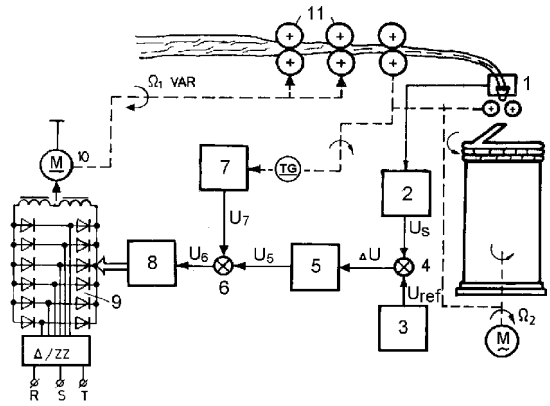


FIGURE 8.4 Drawing frame with closed-loop automatic control of the sliver linear density.

control. The processes here possess relatively high dynamics, and the overall response times, in general, are within several milliseconds. One of the principles used in that field is illustrated in Figure 8.3. An electrical signal is formed at the output of the transducer (1) under the action of the sensing rollers. This signal is proportional, to some extent, to the linear density of the cotton slivers passing through. The transducer is usually an inductive type with moving short-circuit winding. A high-frequency generator powers it to ensure greater sensibility. The sensor output signal is detected, to the balance emitter repeater, (2) and conformed to the input resistance of the memory device (3). The balance circuit (2) secures minimal influence of the ambient temperature and power voltage on the level of the sensor signal. The memory device holds the signal for the time required by the sliver to reach the drafting zone (9). Both the sensor signal and the speed feedback signal drive the phase pulse block (4) from the tacho-generator. (5) The thyristor drive system varies the speed of the DC motor (M) and thus, the drafting rate of the rollers (9). The electro-magnetic clutch (6) is used to couple the rollers (9) to the basic kinematics of the machine at startup. A time relay (7) is used to power the clutch, thus disconnecting the rollers and switching to variable speed. In this way, speed differences throughout the transition processes of starting and stopping the machine are avoided.

Figure 8.4 shows an example of a closed-loop control system on a textile drawing frame. The sliver linear density is measured in the packing funnel using an active pneumatic sensor or alike. The signal is transformed and conditioned by the circuit (2) and compared to the setpoint value U_{ref}. The latter is controlled manually via the potentiometer (3). The error (ΔU) is processed by the regulator (5) according

© 2001 by CRC Press LLC

to the selected control law usually proportional-integral (PI) or proportional-integral-differential (PID). The output voltage (U5) of the controller is added to the average draft rate voltage (U7) from the tacho-generator (TG). The resulting signal is used to govern the variable-speed drive system in which a high-momentum DC motor (10) controls the speed of the preliminary drafting rollers (11). Here, the synchronization between the variable and constant speeds while starting or stopping the machine is achieved by means of the tacho-generator feedback signal. The proposed closed-loop control system cannot influence short-length waves of irregularity within the textile sliver. This is due to the inevitable transport delay when the material passes the distance between the variable-speed zone and the measuring point. To avoid oscillating behavior of the system, some restrictions must be implemented. The most important restriction is to filter and respond to only those irregularities which are at least twice as long as the dead-zone, and whose behavior in the next several lengths can be predicted (extrapolated). This task requires more sophisticated algorithms of the controller than the usual PID techniques.

In an effort to overcome the disadvantages of the mentioned classic controlling techniques, different kinds of combined-type control systems have been implemented in the recent years. Two main problems however, still exist here. The first one concerns the transducers for measuring the linear density of the textile sliver. There still has not been found a method and means for reliable, repeatable measurement of this most important technological parameter. The second problem concerns the high dynamics of the process, requiring development and implementation of new, fast, and accurate devices for real-time control.

Automation of Transport Operations in Spinning Technology

Transporting operations are another important field in which automated systems can be implemented with great efficiency. Figure 8.5 illustrates an approach to building fully automated production line for cards (3) and drawing frames (4). One or several robocars (1) are used to transport the cans with textile slivers. An onboard microprocessor unit controls each robocar. One of its tasks is to trace the path line (2) of fixed type. Transportation paths are scheduled and programmed by the central computer, which also optimizes the routes. The robocars handle the empty and full cans to and from the machines, following the production plan for different mixtures of materials. The operator or worker can call each robocar manually, from each one of the machines which causes rescheduling of the route table by the main computing unit. Figure 8.6 shows the mode of action of a single robocar (1). The can is manipulated by means of the levers (5) which are operated by the onboard control device of the robocar. After the robocar is positioned against the

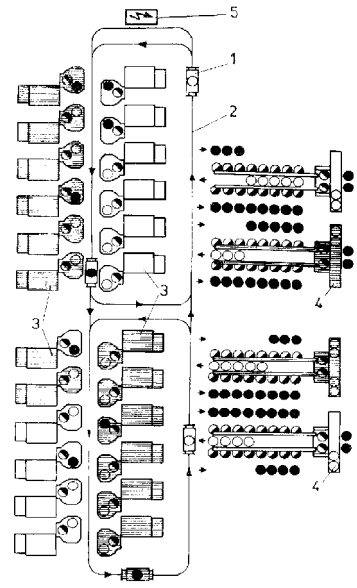


FIGURE 8.5 Automated interfactory transport system for cards and drawing frames.

© 2001 by CRC Press LLC

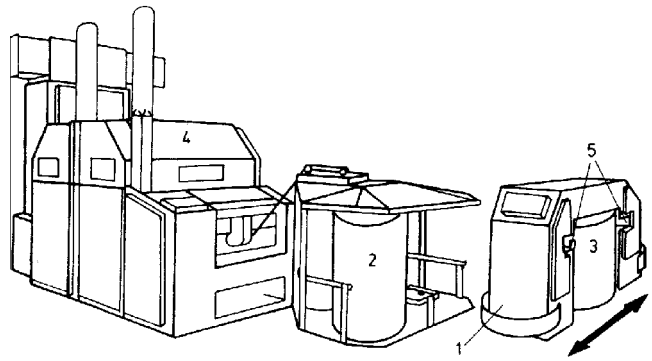


FIGURE 8.6 Robocar system.

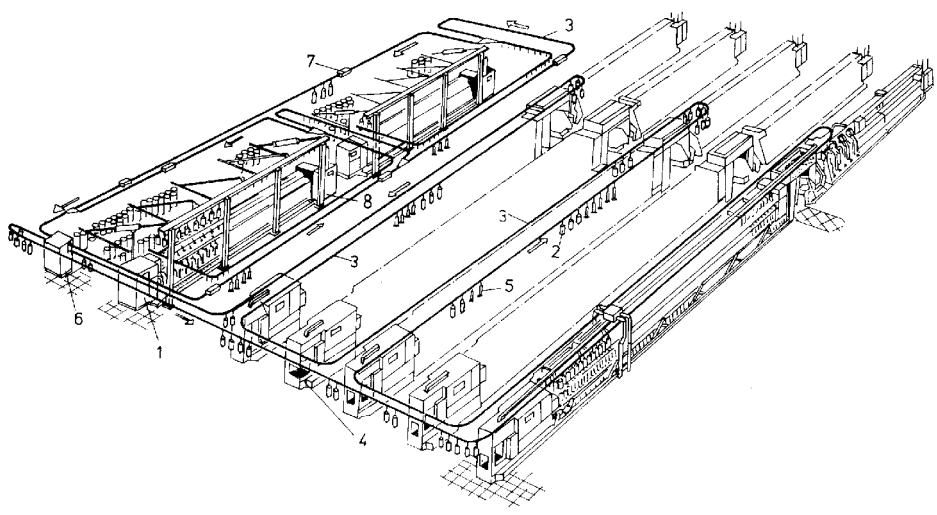


FIGURE 8.7 General scheme of an automated system for feeding of roving and spinning frames.

automatic can changer of the textile machine, the levers (5) exchange the full and empty cans. The empty can is then transported to the previous processing section, e.g., cards, and put in any free place.

Automated Spinning and Post-Processing of Yarns

The spinning and post-processing (doubling, twisting, and winding) of yarns involve machine services by the personnel in which monotonous manual operations are required. A worker operating a ring spinning frame walks an average distance of 10–15 miles per shift while performing manipulations like: changing roving bobbins, binding broken threads, cleaning flying fibers from the drawing assemblies, changing full cops with empty, etc. Some of these manipulations require high skills, and even the most qualified workers cannot efficiently serve the modern high-speed machines. The basic directions for automation of these operations include: design of assemblies to automate the feeding of the spinning frames with roving and cops, automatic exchange and arrangement of full and empty cops, automatic binding of broken threads, automatic cleaning of the machines, aggregating the machines into production lines, etc.

A basic scheme of an automated system for feeding the roving and spinning frames is shown in Figure 8.7. The condenser bobbins (2), obtained from the roving frames (1) are moved through the elevated transport line (3) towards the spinning frames (4). The empty cops (5) are returned to the roving frames by the same transporting facility.

© 2001 by CRC Press LLC

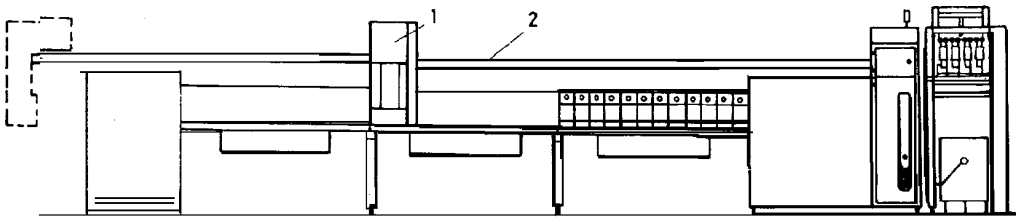


FIGURE 8.8 Spindleless spinning frame served by a robot.

The substantial rate of thread breaks is a characteristic feature of the spinning process. With ring spinning frames, this rate sometimes equals up to 200 breaks per 1000 spindles per hour; thus, the overall productivity of the machine can be considerably reduced, even if a highly qualified worker operates it. With modern high-speed spinning frames (both ring and spindleless) the only way to increase productivity and reduce machine stops is to implement automated techniques. Figure 8.8 shows an approach in implementation of a robot (1) in a spindleless spinning frame. The robot moves alongside the machine on a railroad (2) attached to the machine. Its basic function is to serve as spinning starter while, at the same time, it cleans the rotors from the flying threads. The spinning starter is controlled by a local microcomputer device, which synchronizes the motion of all assemblies. The spinning starter uses a contactless method to control the state of every spinning head of the frame. If a thread break is encountered, the robot is positioned against the head and performs the following manipulations:

It plugs itself into the pneumo-system of the machine and cleans the working place using an arm. The arm is first stretched ahead, and then it moves the motionless rotor and cleans it using a brush and knife while blowing air into the head.
It searches, finds, and gets control on the bobbin thread end and leads it to the zone where the thread is prepared for spinning start.
It brings the prepared thread end to the threading tube and threads it, following the rotor direction after the rotor has been brought into motion.
It handles the processed thread to the winding mechanism of the spinning node.

In case of failure, the manipulations listed above are repeated twice before the spinning node is switched off. The robot [Baumgarter et al., 1989] has an inbuilt microprocessor control unit, which is accessible through the LAN; thus, different modes of action of the robot can be set, e.g., to modes like threader, cleaner, or both. Operating parameters like linear density, yarn twist, staple length of the fibers, rotor diameters, angular speeds, etc., can be set automatically or manually from remote sites like operator's stations of the WAN.

The highest level of implementing automated techniques is reached with the winding textile machines. In the last two decades, durations of hand operations like unloading empty cops, exchanging and arranging ready bobbins, ends binding, etc., have been reduced by more than 15 times by automated systems. Figure 8.9 shows an automated winder. The winding section (1) of the machine is connected to the reserve trunk (2), which is loaded through the feeding box (3).

The level of automation is substantially increased if the spinning frames are aggregated with the automated winders. The productivity rates of these two machines are equal, eliminating stops of the process as a whole; thus, the following advantages are achieved:

Transportation of full bobbins from the spinning frame to the winder is avoided, as well as the cleaning and arranging of the cops and their transportation back to the spinning frame;
Durations of the following preparatory and final operations are reduced: manipulations of the empty cops, placing the roving bobbins in the winder, cleaning the cops, taking off the bobbins, and placing the perns in the winding heads.
Yarn damages are avoided due to the elimination of transport operations.

© 2001 by CRC Press LLC

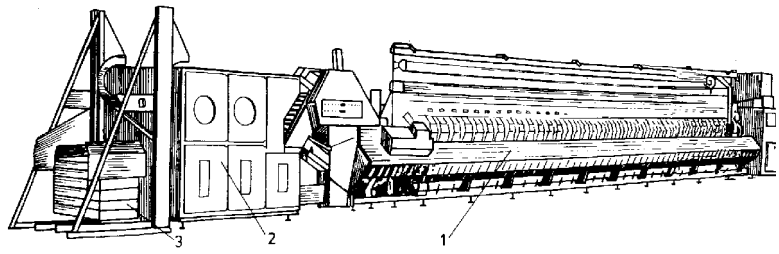


FIGURE 8.9 Fully automated winding machine.

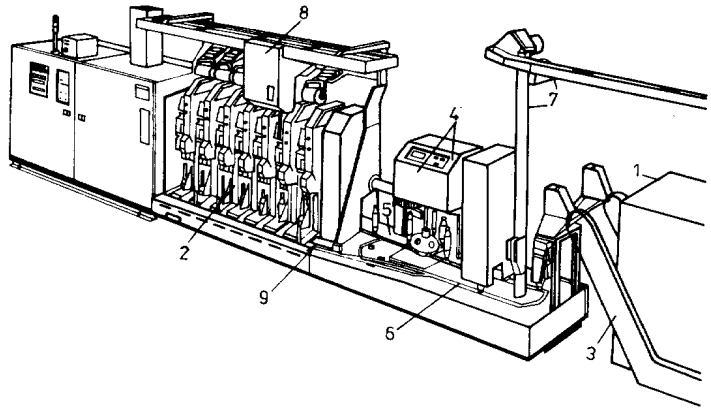


FIGURE 8.10 Aggregating a winder with a spinning frame.

Figure 8.10 shows part of a spinning frame (1) aggregated with an automatic winding machine (2). The spinning frame is equipped with a stationary changer. The full cops are transported from the spinning frame to the winder by means of the transport line (3). They are then stored in the box (4) and, after that, distributed to the winder's heads. If all the heads are busy, the outcoming cops are transported back through the line (6). The empty cops from the winding heads are sent into the trunk of the spinning frame's automatic changer by means of the transporting device (7). The full bobbins are taken from the winding heads by the changer (8). In order to equalize the productivity of the aggregated machines, an additional place (9) is reserved if more winding heads are to be added.

Automated Systems in Weaving

Sizing of Textile Materials

The mixtures for sizing of textile materials are prepared in automated sizing departments (sizing kitchens) containing batch control systems for recipes handling. The controlled parameters in this case are most often temperature, pressure, and time intervals for the preparation of the size. Figure 8.11 shows an example of a fully automated sizing department. Some components are transported using a moving vat (1) through the pipe (2) into the reservoirs (4). The rest of the components (3) are loaded into the installation directly from shipped plastic barrels. For every particular recipe held in the non-volatile memory of the controllers (10) or (11), the components are directed using the distributor (5) to the weighing system (6). From there, the components are fed into the autoclave (9), where they are mixed with water from the pipe (7) and heated using the steam-pipe (8). The sequence is controlled by a microcomputer where the batch program is implemented. The ready mixtures are held in the reservoirs (13) for feeding the sizing machines (14).

© 2001 by CRC Press LLC

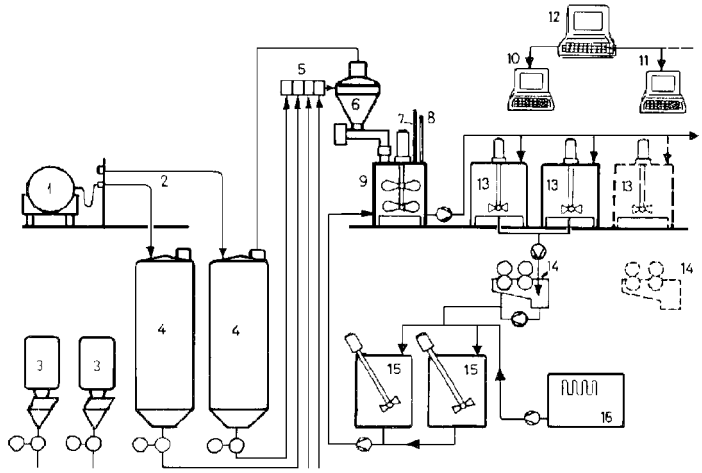


FIGURE 8.11 Automated sizing department.

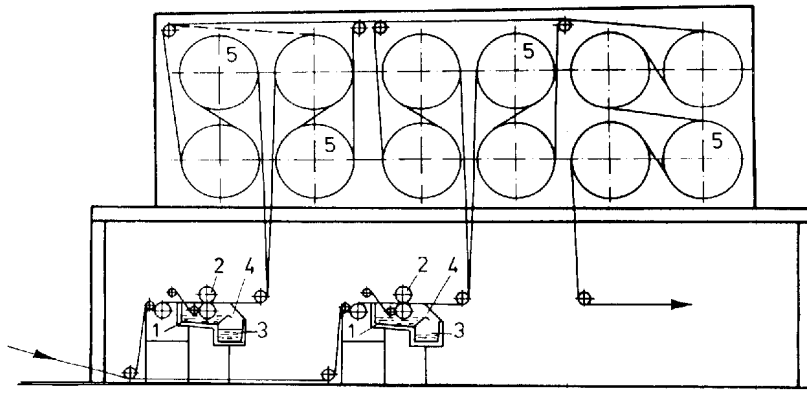


FIGURE 8.12 Sizing machine.

The filtering installation (16) is used to recycle the used size. The station (11) controls the sizing machines, while the microcomputer (12) is used at the higher level to synchronize the requests from the sizing machines and control the sizing department as a whole. Figure 8.12 shows the schematic of a sizing machine.

The main controlled parameters here are the size level, concentration, and temperature in the sizing tub. The level is regulated using the backup tub (3), the overflow (4), and a circulation pump. Constant concentration and viscosity are maintained by adding fresh size in the sizing tub (1). The temperature is controlled by means of a steam heating system. Constant stretch between the transporting and drying drums (5) is maintained by individual variable-speed drive systems. Individual or common heating control is also implemented throughout the process.

Automated Looms

The development of modern control system techniques also concerns such basic textile machines as the shuttleless looms (rapier, gripper, and pneumatic). Modern looms make use of distributed DC and AC drive systems, synchronized by a central control unit. Figure 8.13 shows the structure scheme of such a system implemented for a rapier textile loom. The position of the individual working assemblies is controlled by different sensor systems.

© 2001 by CRC Press LLC

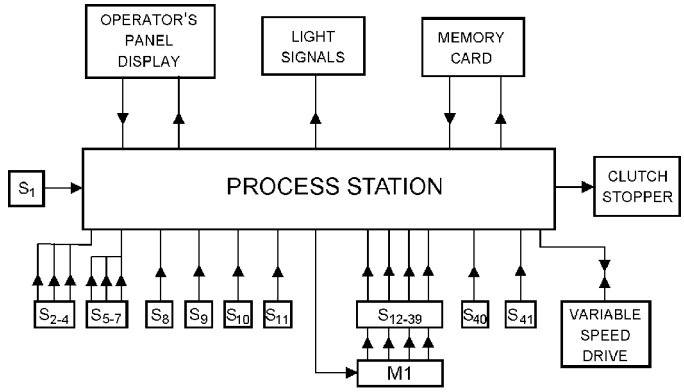


FIGURE 8.13 Structure scheme of a loom microcomputer controller.

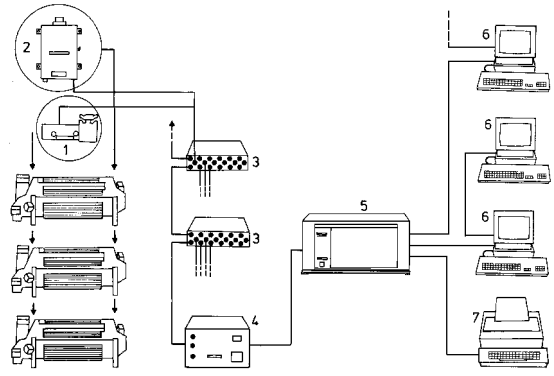


FIGURE 8.14 Distributed data acquisition and control of looms.

The angular position of the main loom shaft is detected by the sensor (S1) with a resolution greater than one angular degree.

The position of the rapiers (left and right) is registered using the sensors (S2) through (S7) (three sensors for each rapier).

The system implements a self-learning algorithm in the following way: when the loom is started for the first time, or after repair, the main shaft is rotated at least twice and the angular positions of the shaft, corresponding to at least three positions of each rapier, are stored in memory. After that, the normal operating mode is started. During its normal operation, the control system constantly compares the positions of the rapiers in these pre-registered points with the remembered values. The loom is automatically stopped if the deviations go beyond certain limits set by the operating personnel.

Breaks of pick threads are registered using the sensor (S8). The loom is automatically stopped after the main shaft is rotated to a preset angle, which is also changeable through the operator's keyboard. Several sensors (S9) are used to control breaks of warp threads in several warp zones of the loom. The individual heald sensors (S12) through (S39) control the heald frames positions. The heald machine program is stored and retrieved from the computer memory, and used to control the current weave formation. Changes of the weave can be made either manually, using the computer man-machine interface, or by means of external data carriers; i.e., disks, EEPROM memory modules, etc. The sensor (S40) measures the length of the produced fabric. Applying integrated control systems plays an essential role in efforts to increase the productivity of the weaving process and achieve better fabric quality. Integrating loom controllers of the above described kind leads to structures, an example of which is presented in Figure 8.14.

© 2001 by CRC Press LLC

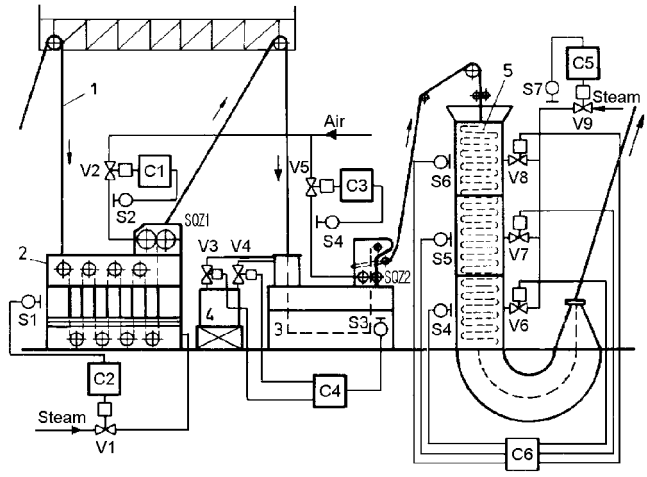


FIGURE 8.15 Automated bleaching line.

Individual operator's units with keypad and display are mounted on each loom, in addition to the set of sensors described above. Up to 15 kinds of stops, e.g., due to machine malfunction or repair, are registered automatically. The productivity and other performance parameters of the looms are also measured or calculated. Data concentrators (3) are used for every cluster of 8/16 looms. The bus interface unit (5) serves as a gateway, connecting the local network at process level with the factory information network at the higher level where the information can be accessed and managed using the stations (6).

Automated Systems in Finishing

Finishing incorporates complex chemical processes which require precise control of a variety of parameters: level of water, chemical solutions and compounds, temperature of the materials, humidity of the fabrics after drying, acid and base concentrations, hydrogen ions activity, etc. This stage of textile processing requires sophisticated automated systems and implementation of modern control methods.

Bleaching of Textile Materials

Figure 8.15 shows an example of an automated bleaching line. The cloth (1) enters the washing machine (2) where constant temperature is maintained by the single-loop system of the sensor (S1) and the controller (C2). The steam flow for heating is controlled through the valve (V1). The squeezing rollers (SQZ1) at the output of the washing machine, decrease the water in the material. A pneumatic system is used to maintain constant pressing force to the rollers, an important condition for the process. The controller (C1) measures the pressure using the sensor (S2) and controls it by means of the valve (V2). An automatic system for water level control is often implemented here. From the washing machine, the cloth is transported to the saturator (3), where it is emerged in sodium base to remove the remaining size and other particles. Maintaining constant concentration of the solution is most important at this stage of the process. This task is accomplished by the controller (C4). The sensor (S3) measures the inside concentration. The controller actuates the valves (V3) and (V4), which change the flow of water and concentrated solution from the dosing system (4) to the saturator (3). At the output the water content of the fabric is reduced by means of the squeezing rollers (SQZ2). The pressing force of the rollers is kept constant by means of a control system comprising a controller (C3), a sensor (S4), and a valve (V5). From the saturator, the material is passed to the last section—the compensator (5). Three temperature zones are controlled using the sensors (S4), (S5) and the controller (C6). The controller (C5) maintains constant pressure of the heating steam.

Sometimes the process of bleaching is intensified by using higher pressure and temperature. An example of such an apparatus is shown in Figure 8.16. The controller (4) is used to maintain constant pressure of the steam inside the volume. The transducer (5) forms a signal proportional to the difference

© 2001 by CRC Press LLC

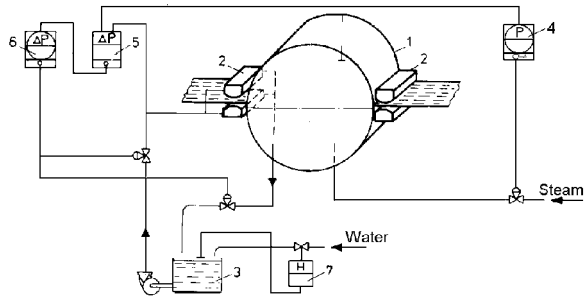


FIGURE 8.16 Automated system for boiling and bleaching of textile materials.

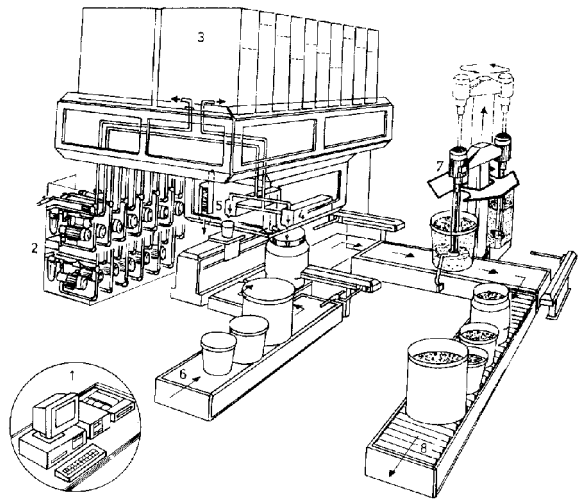


FIGURE 8.17 Automated department for dye-stuff preparation.

between the pressures of the input and the output boxes (2). The controller (6) maintains this pressure difference in the range of 0.26–0.30 KPa, to avoid fast wearing of the contacting surfaces (2).

Automated Systems in Dyeing

Automated systems in dyeing help producers to decrease wastes of expensive chemicals and to react adequately to the market demands. The basic problems here require use of colorimetric methods for recipe preparation and handling. Colors should exactly match the patterns, and the system should be able to reproduce them on demand. Using colorimetric methods for recipe calculation ensures high color matching precision and cuts recipe preparation time. These methods are based on the existing dependence between dye concentration and the spectral reflection coefficient of the sample. The optical characteristics of the samples are measured using specialized spectro-photometers. The data obtained is then used to calculate recipes for reproduction of the chosen color on the basis of a group of dyeing chemicals. Colorimetric systems can increase the utilization of chemicals up to 10-20%, and provide reliable, online control of the dyeing process with less interventions for recipe corrections. Figure 8.17 shows a typical automated dyeing department. The dyeing components are kept in the reservoirs (3). Each component is transported to the reservoir using a pump and a filter as shown in, part two of Figure 8.17. The components are periodically recirculated to prevent settling down. The dosing block (4) incorporates separate valves for each component. The valves are regulated in increments at four levels or continuously. The valves are operated by the microcomputer control unit (1), which serves as a batch controller. The components are fed into the transporting receptacles (6) by the dosing section (4) on the line (6). An automated weighing system controls

© 2001 by CRC Press LLC

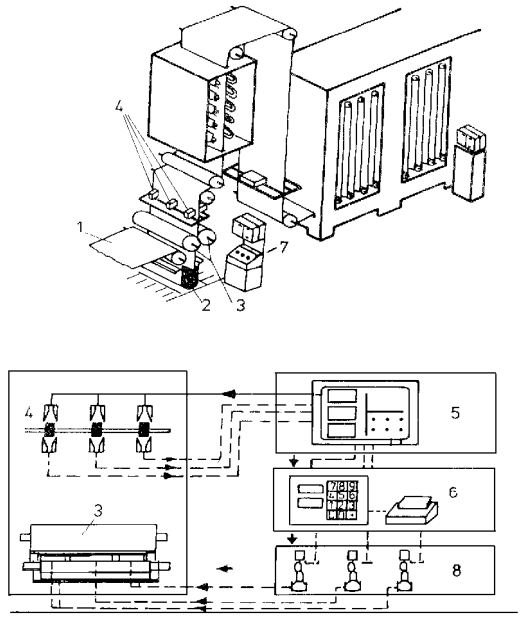


FIGURE 8.18 Automated machine for continuous dyeing.

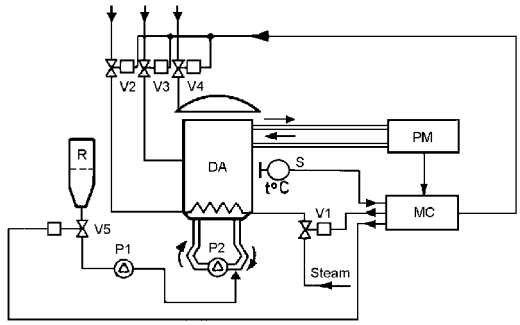


FIGURE 8.19 A dyeing apparatus with dye-stuff extraction control.

the dosing process, based on the current recipe. A stirring device with variable speed homogenizes the contents of every vessel, and the ready components are transported out through the roller line (8). The system also utilizes an additional block for color sampling used for testing purposes.

With modern dyeing machines, equal coloring of the fabrics is achieved by controlling the pressing force of the squeezing foulard rollers. This is done with respect to the humidity of the processed material. Figure 8.18 shows part of an apparatus for continuous dyeing with the system for automatic control of material humidity after the foulard. The automation of dyeing concerns mainly optimal control of the process of extracting the dyeing substance. This task is accomplished by controlling such parameters as: solutions concentrations, temperature, and pH. Figure 8.19 shows the principle scheme of a dyeing system. The process takes place in the dyeing apparatus (DA). The dye-stuff is supplied from the reservoir (R) through the valve (V5) and the pump (P1). The pump (P2) is used for recirculation, while the photometer (PM) measures the current concentration of the solution. The regulating valve (V1) controls the heating steam flow. The sensor (S) detects the current temperature of the solution. Water and additional chemicals are supplied through the valves (V2), (V3) and (V4). High dyeing quality is achieved when the process follows a predefined curve of dye extraction. The process is monitored by the microcomputer control unit

© 2001 by CRC Press LLC

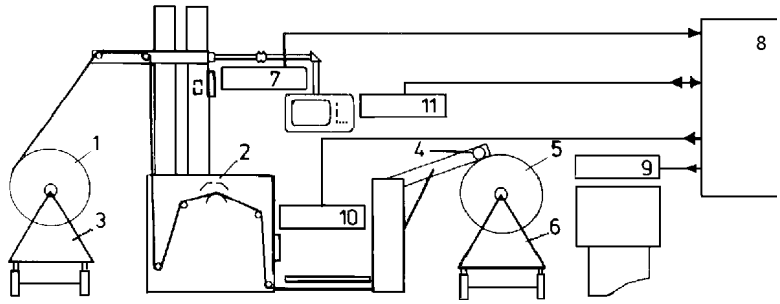


FIGURE 8.20 Final control of cloth quality by means of an automated inspecting machine.

(MC), where the current values of the concentrations are compared against the predefined ones. The rate of dye extraction is controlled by means of the steam flow and the flow of additional chemicals to the apparatus.

Automating the Final Quality Control of Cloths

Modern automatic quality control systems are also implemented in the final stage of the textile production, where the quality of the ready fabrics is monitored before they are shipped to the customers. Visual control of the cloth quality, usually performed by man, is difficult to replace with appropriate automatic device because of the specific requirements for this kind of control. Several approaches to resolving this problem have found industrial application in modern textile technology. An example of such a quality control system is shown in Figure 8.20. The cloth formed in a roll is transported to the inspecting machine (2). After it has passed through the controlling device, the cloth is rolled again and the roller (5) is transported to the warehouse using the carriage (6). A video camera (7) moving alongside the roller's width is used to monitor the cloth surface. The video signal is converted into digital form and is stored dynamically in the microcomputer (8) memory. The video display unit (11) is used for both visual control of the surface by the operator and for displaying alphanumerical and graphical information from the computer; thus, one of the narrow places in quality control is avoided, resulting in an output speed of the controlled fabric to more than 100 m/min.

8.3 Distributed Systems for On-Line Quality and Production Control in Textiles

Basic Concepts for On-line Control in Textiles

At present, the majority of conventional (manual) methods for quality control are being replaced by automated systems. The times when the laboratory staff circulated the site every two hours, cut samples of sliver or yarn, and then weighed them manually or passed them through a stand-alone quality controlling device, now belong to the past. In the early 1970s, there came to exist ideas for automated systems, aimed at replacing human labor in this challenging field. Figure 8.21 gives an example of the ideas behind the first realizations. In this example, a central processing unit (3) collects information from four different sources: the opening department, the cards, the drawing frames, and the spinning machines. Each local unit (1) and (2) serves as a database holder and was developed as a dedicated (embedded) system. The information concerns such parameters as machine stops, productivity/quantities, linear density, long- and short-wave regularity, spectrograms, etc. Figure 8.22 gives a more detailed look at the structure of a typical first generation, automated quality control system in cotton spinning (14). The data processing units of type (1) are dedicated microprocessor controllers, which serve as data collection and concentration. The units of type (2) perform more complex calculations like statistic parameters, distributions, linear density diagrams, and spectrograms. Although, at this stage of industrial automation

© 2001 by CRC Press LLC

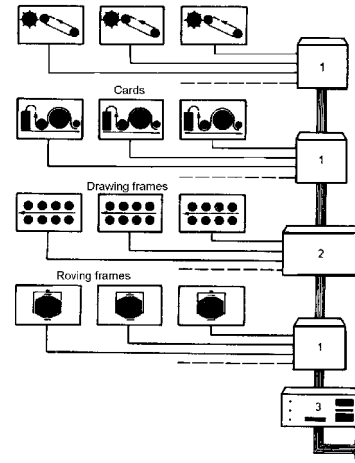


FIGURE 8.21 Structure of a first-generation data acquisition and control system in textiles.

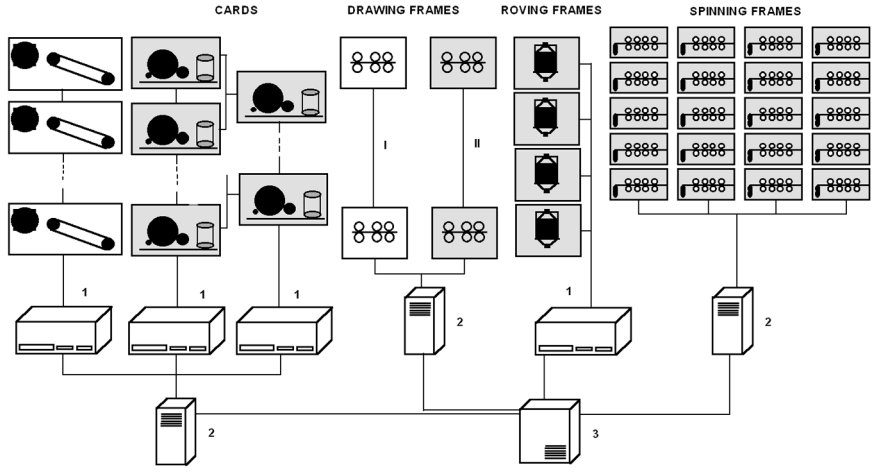


FIGURE 8.22 Basic structure of an automated production and quality control system.

there have not yet been adopted layer definitions and standards, some level of diversification can be observed here. Central processing units of type (3) serve as database holders and information management stations at factory level. With the further development of microcomputers after 1985, dedicated industrial automation systems began to be replaced by devices implementing market industrial versions of PC hardware and software. This is not the case in textiles, regarding devices of type (2), because the quality control parameters and, especially, the spectral characteristics, still require dedicated hardware.

Approaches to Building Cost Effective Real-Time Control Systems in Textiles

Real-Time Process Monitoring

The dispersive nature of textile technology, with hundreds of parameters to measure and control, often brings the need for implementation of relatively complex and expensive automated systems. The processing of textile slivers is a typical case. Figure 8.23 presents a common process flow diagram for the classical card cotton spinning. The first passage double-head drawing frames (1DF1-1DF10) receive textile slivers from the previous machine array of cards and are followed by a second-passage drawing frames, (2DF1-2DF10). In this case, there are 160 input and 40 output slivers running at a speed of 5-10 m/s. Production and/or quality control require that their linear density should be controlled on-line, at least at 40 points, in order

© 2001 by CRC Press LLC

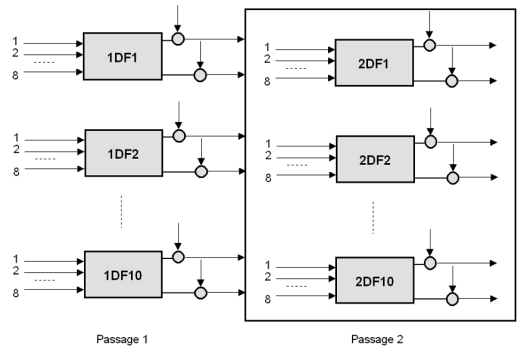


FIGURE 8.23 Common process flow diagram for card cotton spinning. (From Djiev, S.N. and Pavlov, L.I., 1995, Low-cost automated system for on-line quality and production control in textiles, in *Proc. Control. Qual.*, (7), 179–183. With permission from Elsevier Science.)

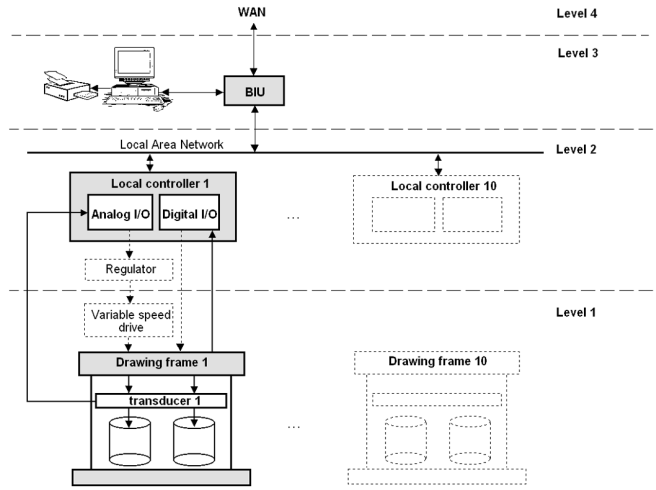


FIGURE 8.24 Hierarchy in distributed quality and production control systems. (From Djiev, S.N. and Pavlov, L.I., 1995, Low-cost automated system for on-line quality and production control in textiles, in *Proc. Control. Qual.*, (7), 179–183. With permission from Elsevier Science.)

to ensure supply of the following processing sections with material of desired quality. Some parameters like spectra, diagrams, control charts, etc. will have to be calculated, too. In addition, start/stop time intervals for each machine/worker should be registered and classified, as well as summary statistics for the production site as a whole. In this and similar cases, multi-purpose automation systems offered by the market cannot meet the specific quality control requirements set forth by the textile technology. One solution is to implement a smaller, dedicated system with flexible and open structure, allowing interconnection and further integration. An example of such a system is shown in Figure 8.24. At the lowest Level 1, intelligent field sensors provide standard signal, or communicate using the network, carrying information for the linear density of the drawing frame slivers. Start/stop intervals and productivity are measured, as well. Local controllers at Level 2 serve as data acquisition units with both analog and digital inputs. Basic statistics, spectra, and quality charts related to the quality control requirements in textiles are obtained on-line. A PC-based operator's station at Level 3 controls the system via the bus interface unit (BIU) and communicates with the local controllers. At preset time intervals, or on request, the station collects information from one or all controllers and refreshes its internal data base, which is used to produce data sheets of various types, including quality charts, and report on an hourly, daily, or monthly basis.

© 2001 by CRC Press LLC

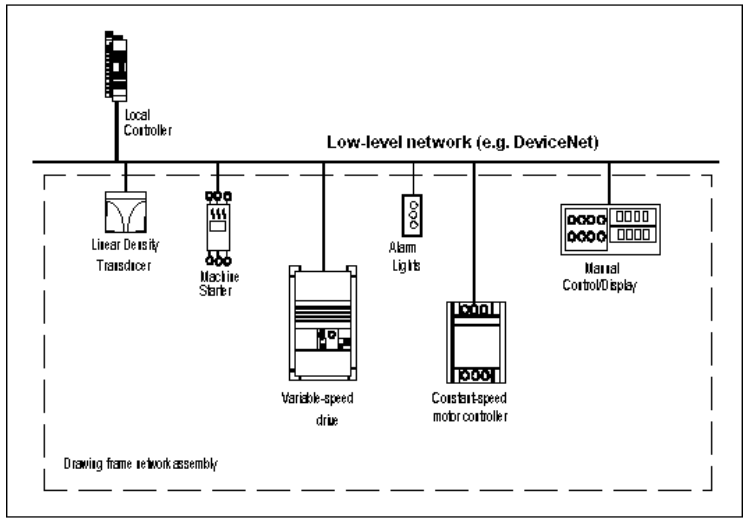


FIGURE 8.25 Field-level network with intelligent end devices.

Alarms are activated and/or machines are stopped if the linear density of the sliver goes beyond certain limits during a preset time interval. This structure allows further development of the system towards distributed feedback control of the drawing frames using the analog outputs of the local controllers and variable-speed drives. From the network side, the control system field devices (network nodes of a single drawing frame) take the form presented in Figure 8.25.

On-Line and Off-Line Quality Control of Textile Materials

In many cases with textile processes, the production quality is controlled by means of spectral analysis of the output material, linear density of which is a basic quality parameter. The processed materials, textile slivers, are passed through roller drafters to obtain desired cross-sectional area (thickness) and linear density of the resulting product. It is an immanent property of these processes that they do not serve as low-pass filters for the disturbances, as is the case with most of the other technological processes in industry [Djiev, 1994]. As a result, regardless of whether the process is regulated or not, it never reaches 'steady state' due to the frequent disturbances of substantial magnitude. The latter are caused mainly by the variations and/or dimensions of the input material properties. In some cases of machine kinematics malfunction, the process itself may generate waves of irregularity.

The final estimate of the product quality cannot, therefore, be based only on elementary statistics or control charts because the variations inside the samples are also of great importance. On the other hand, the current values of the monitored quality parameter, in this case linear density, are practically unusable as raw data. This is due to the great output speeds values (5-15 m/s) and the large number of control points at the production site. For these reasons, an important part of the quality control procedures becomes the evaluation of the spectral density $S_y(\omega)$:

$$\int_{-\infty}^{+\infty} S_y(\omega)\, d\omega \;=\; \lim_{T\to\infty} \frac{1}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} |y(t)|^2 \, dt \tag{8.1}$$

where, $y(t)$ = output linear density, T = period [s], and ω = frequency [rad/s].

The precise value of $S_y(\omega)$ cannot be evaluated because of the finite bandwidth $\Delta\omega$ of the filters used. What can be controlled, in practice, is the average power of the signal for a given frequency band $\omega_0 \pm \frac{\Delta\omega}{2}$

$$S_y(\omega_0, \Delta\omega) \;=\; \frac{1}{\Delta\omega} \int_{\omega_0 - \frac{\Delta\omega}{2}}^{\omega_0 + \frac{\Delta\omega}{2}} \int S_y(\omega)\, d\omega \;=\; \frac{1}{T} \int_0^T |y_F(t)|^2 dt \tag{8.2}$$

© 2001 by CRC Press LLC

where, $y_F(t)$ is the controlled signal passed through a filter with rectangular amplitude-frequency response in the range $\omega_0 \pm \frac{\Delta\omega}{2}$. This section briefly discusses approaches to on-line and off-line process control quality using spectral criteria and their implementation in the textile industry.

Spectral Criterion

Consider a drafting process with input linear density $x(t)$ and output linear density $y(t)$ of the textile sliver. A spectral criterion (SC) can be used here to measure the distance between the real spectrum obtained from the process and the so-called normal spectrum

$$SC = \int_{\omega_{\min}}^{\omega_{\max}} \{|W_s(j\omega)|^2 S_x(\omega) - [\bar{S}_{\text{ideal}}(\omega) + \alpha(\omega)]\}\, d\omega \tag{8.3}$$

where,

$W_s(j\omega)$ = complex gain of the drafting process as a dynamic system (with or without automatic control) through the main (load) channel $x(t) \rightarrow y(t)$,

$S_x(\omega)$ = spectrum of the input textile sliver (process load), and

$S_{\text{ideal}}(\omega)$ = desired spectrum density of the output sliver as determined by the theory of the process. In some cases, the ideal spectrum may be obtained in analytical form. For example, in textile drafting the ideal spectrum for cotton slivers containing fibers of equal length L_F is

$$S_{\text{ideal}}(\omega) = \frac{2}{\pi L_F \omega^2}(1 - \cos \omega L_F) \tag{8.4}$$

The additive correction $\alpha(\omega)$ here is used to form the normal spectrum curve, and is associated with the inevitable influence of the nonideal conditions on the process, including nonlinear phenomena, uncontrolled disturbances, etc.

Formally, the value of the SC equaled the area with boundaries determined by the lines of the real and the normal spectrum of the textile sliver. A graphic interpretation of the *SC* is presented in Figure 8.26. The integral boundaries in Equation 8.3 are different for the sections A, B, C, and D, all shown in Figure 8.26.

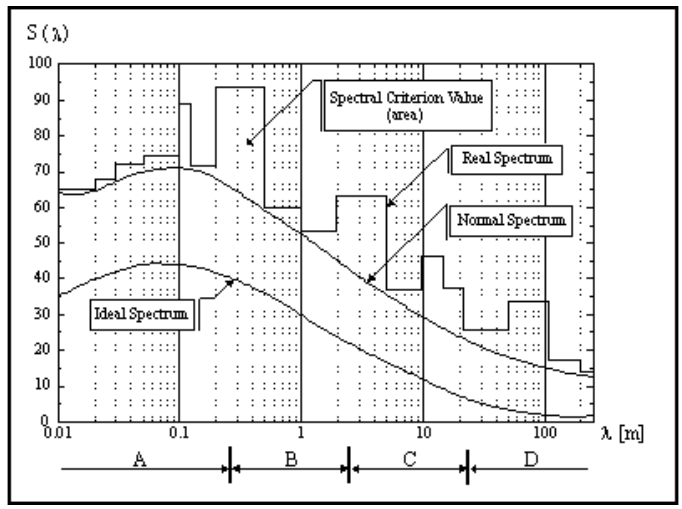


FIGURE 8.26 Graphic interpretation of the spectral criterion. (From Djiev, S.N., 1997, Automated quality control using spectral criteria, in *Proc. Control. Qual.*, 9(1–3), 17–21. With permission from Elsevier Science.)

© 2001 by CRC Press LLC

Off-Line Estimation of the Spectral Criterion

At present, automated quality control usually deals with time series (realizations of the controlled linear density $y(t)$) [Djiev, 1997]. If, for example, $N(N = N_S.N_K)$ values have been recorded with sample time T_S, then an estimate of the spectrum, using Fourier transform is given by

$$S(lg\lambda) = \frac{\ln 10}{\lambda \bar{y}^2 N_s} \sum_{j=1}^{N_s} \left\{ \left| \sum_{i=1}^{N_k} (y_i - \bar{y}) \exp\left[-j2\pi \frac{i + (j-1)N_k}{\lambda} \right] \right|^2 \right\} \tag{8.5}$$

where, $\bar{y}$ = mean value of $y(t)$ for the period $T = N_K.T_S$, $\lambda = \frac{2\pi v}{\omega}$ = wavelength at output linear speed v of the textile sliver.

Smooth spectra are obtained by averaging, as in Equation 8.5, or by means of windows applied either to the time series or to the spectrum itself [Papoulis, 1973]. As it previously stated, any attempt to execute this task in real-time mode of the quality control systems is obstacled by the requirements for speed and memory that cannot be met by the commonly used industrial computing devices.

On-Line Estimation of the Spectral Criterion

Hardware Realization

Estimates of the spectrum can be obtained using a system of n parallel bandpass filters with fixed parameters, each one tuned to a central frequency ω_i. The signal is processed following the algorithm from Equation 8.2.

$$y(t) \rightarrow \Pi\text{-filter} \rightarrow (\ )^2 \rightarrow \frac{1}{T}\int_0^T \rightarrow S_y(\omega_i, \Delta\omega), \quad i = [1,n]$$

In this case, a subsequent analog-to-digital conversion is needed for the output signals of all channels in order to integrate the spectrum analyzer in a computerized quality control system. The period T here is critical for the precision of the estimates [Max, 1981]. Typical problems here are the unstable filters frequencies ω_i, and the variations of their gain and bandwidth values. This approach is used when designing stand-alone quality control devices, working off-line.

Software Realization

The commonly used software approach is based on different modifications of the fast fourier transform (FFT), or the Hartley transform [Baumgarter et al., 1989; Djiev, 1997]. The main disadvantage concerning quality control, is the fixed number of spectrum (frequency) points of the FFT. Although interpolation can be used for the visual presentation of the spectra [Papoulis, 1973] it is impossible to satisfy the needs of the real-time quality control because the cycle time is usually very limited. On the other hand, on-line parallel pipeline FFT techniques are too expensive to be widely implemented in industrial quality control because they request sophisticated software and hardware [Petrovsky, 1988].

Another approach makes use of an optimal system of parallel digital bandpass filters [Djiev, 1997; Petrovsky, 1988; Wong and Andres, 1993]. Each one of the filters can be presented with its differential equation in the form of a 'continuous' transfer function with differential operator s:

$$W_F(s) = \frac{y_F(s)}{y(s)} = \frac{\Delta\omega.s}{s^2 + \Delta\omega.s + \omega_0^2}, \tag{8.6}$$

or as a discrete function in the z-plane (z^{-K} representing a delay of k sample steps) [Djiev, 1997]:

$$W_F(z) = \frac{y_F(z)}{y(z)} = k_1 \frac{1 - z^{-2}}{1 + (k_1 - 1)k_2 z^{-1} + (1 - 2k_1)z^{-2}}, \tag{8.7}$$

© 2001 by CRC Press LLC

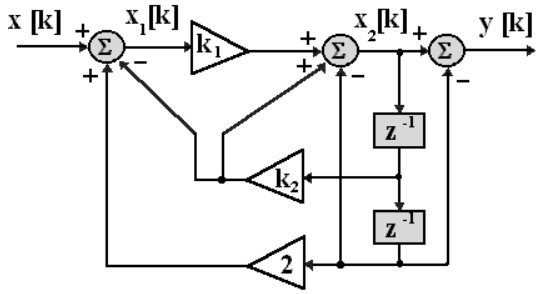


FIGURE 8.27 Structural scheme of a bandpass digital filter for on-line spectra estimation. (From Djiev, S.N., 1997, Automated quality control using spectral criteria, in *Proc. Control. Qual.*, 9(1–3), 17–21. With permission from Elsevier Science.)

where the coefficients

$$k_1 = \frac{\Delta\omega T_s}{2 + \Delta\omega T_s}$$

and

$$k_2 = 2\cos\omega_0 T_s$$

independently determine the bandwidth $\Delta\omega$ and the central frequency ω_0 of the filter with sample period T_S.

Application of the Spectral Criterion in the Quality Control of Textile Slivers

The spectral density can be evaluated using a system of digital filters (described above) with a software realization on a microcomputer process station [Djiev and Pavlov, 1995], or using single-chip digital signal processors at field level [Petrovsky, 1988]. The program module algorithm of the digital filtration [Flatau, 1988] for a given frequency band follows the equations:

$$\begin{aligned} x_1[k] &= x[k] - k_2 x_2[k] + 2x_2[k-2] \\ x_2[k] &= k_1 x_1[k] + k_2 x_2[k-1] - x_2[k-2] \\ y[k] &= x_2[k] - x_2[k-2] \end{aligned} \tag{8.8}$$

where, $x[k]$ is the value of the filter input signal $x(t)$ at the moment $t = kT_S$, $k = 0, 1, 2, \ldots$. The direct programming of each element of the software spectrum analyzer is illustrated in Figure 8.27. In textile quality control, the wavelengths range between 0.01 and 400 m (i.e., $\omega = 0.15 - 6000$ rad/s). For an estimate of the spectral criterion in the entire range (see Figure 8.26) a total of 32 bandwidth filters of the type [Jourdain, 1973] are needed. More often, the complexity of the task is reduced when the textile sliver spectrum is investigated in the sub-ranges A, B, C, and D in Figure 8.26, depending on the type of the technological process.

The limited number of channels (32 max) and the narrow bandwidth make it possible for conventional microprocessor or microcomputer, e.g., PC-based, systems to be used for on-line quality control based on spectral estimates. This approach is usually implemented in low cost automated quality and production control systems [Djiev and Pavlov, 1995], used for monitoring the quality of cotton slivers in textiles.

Integrating Control and Manufacturing Systems in Textiles

When investing in computer controlled/connected textile machines and installations, some considerations must be made in advance. While in the late 1980s the overwhelming majority in that field were single CAD, CAM and computer-aided quality control (CAQ) systems, today efforts should be made toward

© 2001 by CRC Press LLC

integrating these systems in flexible and open structures. The main reasons for this can be found in the following list of disadvantages, which 'island' solutions exhibit:

The different subsystems, i.e., systems for purchase, planning, management on one side and for patterning, weaving, knitting, etc., on the other work on separate databases, although the subsystems themselves may be quite computerized.
Multiple copies of database information exist in each one of these subsystems.
Data actualization becomes costly and ineffective.
Data exchange between the subsystems needs special actions to be taken, most of which are by operators.
It is hard to fulfill the just-in-time production requirements.

From a more global point of view, there are three main challenges for today's textile manufacturers:

To have a good estimate of the current market state.
To offer new products and/or models within short reaction time.
To keep the production process effective enough to be worth investing in.

Integrating the existing CAD/CAM, CAP, and CAQ systems into a computer-integrated industry unit (CAIU) is an important step in that direction. A CAIU, in fact, represents the whole textile factory in which the office, administrative, financial, and personnel management systems integrate their hardware, software, and databases with the rest of the computer aided systems—those for planning, design, and production. As with other industries, the following three fields of activity can be outlined for an automated textile factory: CAD, CAM, and Planning and Production Management (PPM).

Usually, the integrated state of these three subsystems is denoted as CIM. If the computer-aided office (CAO) system is added [Schlotterer, 1989], we obtain a preliminary model of how a CAIU is built (Figure 8.28). A more detailed description of the information and material flows in a CAIU is shown in Figure 8.29. It represents a common CAIU model, adopted by some of the contemporary developers of integrated systems [Baumgarter et al., 1989; Flatau, 1988]. As observed from the information flows in the CAIU, the unit management team determines its goals based on the market demands and state. The quality control, production, and planning systems have actually penetrated all other parts of the CAIU.

8.4 Summary

The world of industrial production, and textiles in particular, will be facing radical changes with the future development of new concepts and technical means for automation. Computer integrated and flexible production systems will substantially change the way people act and participate in the technological

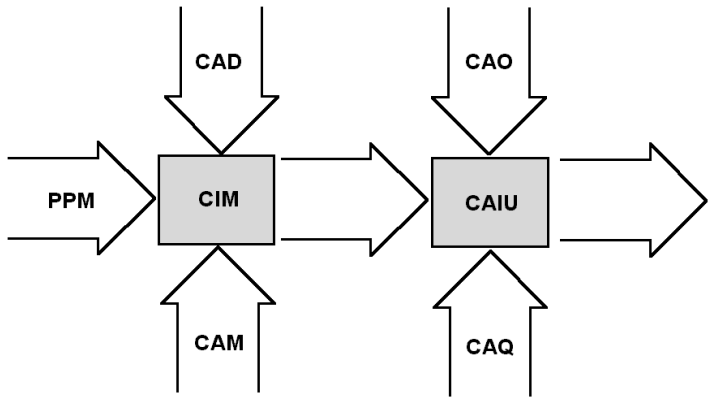


FIGURE 8.28 Integrating computer aided systems in modern manufacturing.

© 2001 by CRC Press LLC

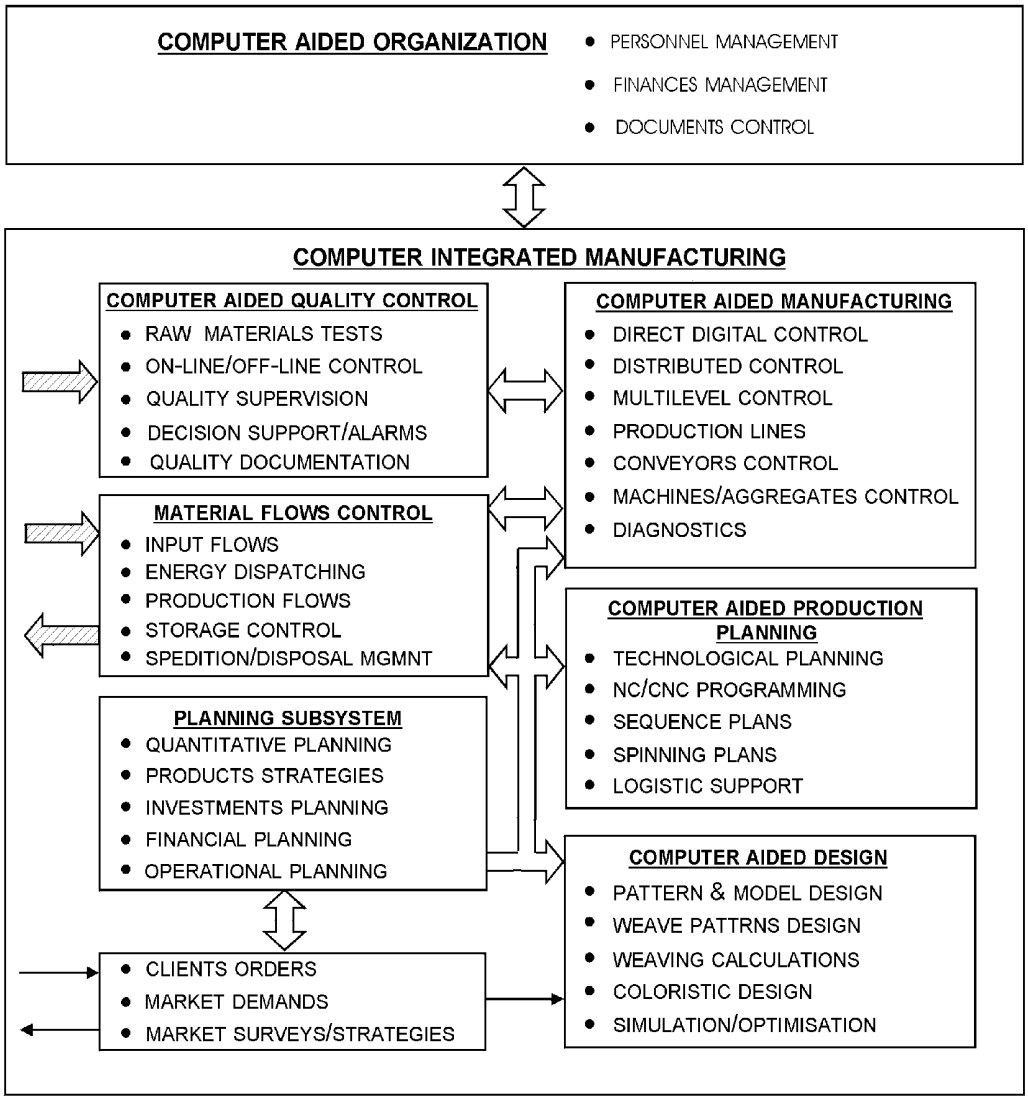


FIGURE 8.29 CIM/CAI model for textile manufacturing.

process. Routine and monotonous service duties are increasingly being replaced by managing, planning, and controlling actions. Because of the specific requirements of textile processing, however, a large number of workers will still be involved at the production site.

Textile entrepreneurs are not so enthusiastic when it comes to large investments, because automation offers both opportunity and risk. Unlike high-tech fields, the textile industry often has a way out to the emerging labor markets as the world becomes more and more open. In the recent decade, it became much more profitable to establish new production facilities abroad than to start expensive programs for production automation and system integration at home. The main reason for this is the great diversification of labor costs around the world. For an optimal solution, both technical and economical considerations need to be taken into account. Here, we leave the purely technical domain and enter a challenging field where only an interdisciplinary approach, combining economics and engineering, will help the textile industry make its way through.

© 2001 by CRC Press LLC

References

Baumgarter, H., Knischewski, K., and Wieding, H., 1989, CIM Basisbetrachtungen, Siemens AG, Nurnberg-Moorenbrunn.

Binder, R., 1987, Automatizierungsmöglichkeiten in der Spinnerei, *Melliand Textilber.*, (1) 527–532.

Djiev, S. N. and Pavlov, L. I., 1995, Low-cost automated system for on-line quality and production control in textiles, *Proc. Control Qual.,* (7), 179–183.

Djiev, S. N., 1997, Automated quality control using spectral criteria, *Proc. Control Qual.,* 9(1–3), 17–21.

Djiev, S. N., 1994, Modeling a double-zone drafter as an object of control, *Text. Res. J.,* 64, 449–456.

Djiev, S. N. and Pavlov, L. I., 1993, Untersuchung der dynamischen Eigenschaften elektromechanischer Meβwertüberträger bei der Feinheitsmessung von Faserbändern, *Melliand Textilber.,* 74, 713–716.

Flatau, U., 1988, CIM Architectural Framework, Digital Equipment Corp., Mainard, MA.

Jourdain, G., 1973, Comparaison de différentes méthodes de reconstitution d'un signal fourni sous forme échantillonnée, *Anales des Télécommunications,* 28(1–2).

Linggard, R. and Boyd, J., 1981, High speed digital filter bank, *IEE Proc.,* 128 (E), 212–216.

Max, J., 1981, Méthodes et techniques de traitement du signal et applications aux mesures physiques, I–*Principes généraux et méthodes classiques,* Masson, Paris.

Papoulis, A., 1973, Minimum bias windows for high-resolution estimates, *IEE Trans.,* 1(19).

Petrovsky, A. A., 1988, Methods and microprocessor systems for real-time processing of wideband and high-speed signals, 68–86, Nauka and Technika Publ., Minsk.

Rembold, U., Nhaji, B. O., and Storr, A., 1993, *Computer Integrated Manufacturing and Engineering*, Addison-Wesley, Reading, MA.

Uster Bulletine, 1985, Uster, (33).

Schlotterer, H., 1989, CAI als Rationalizierungsperspektive in der Textilindustrie, *Melliand Textilber.,* (1).

Wong, R. Y. and Andres, A. L., 1983, Efficient real-time signal processing with digital bandpass filters, *Proc. SPIE: Real Time Signal Process, YI,* 431, 270.

© 2001 by CRC Press LLC